

March 2014

AMIDAS-II: Upgrade of the AMIDAS Package and Website for Direct Dark Matter Detection Experiments and Phenomenology

CHUNG-LIN SHAN

*Physics Division, National Center for Theoretical Sciences
No. 101, Sec. 2, Kuang-Fu Road, Hsinchu City 30013, Taiwan, R.O.C.*

*Department of Physics, Hangzhou Normal University
No. 16, Xuelin Street, Xiasha Higher Education Zone, Hangzhou 310036, Zhejiang, China*

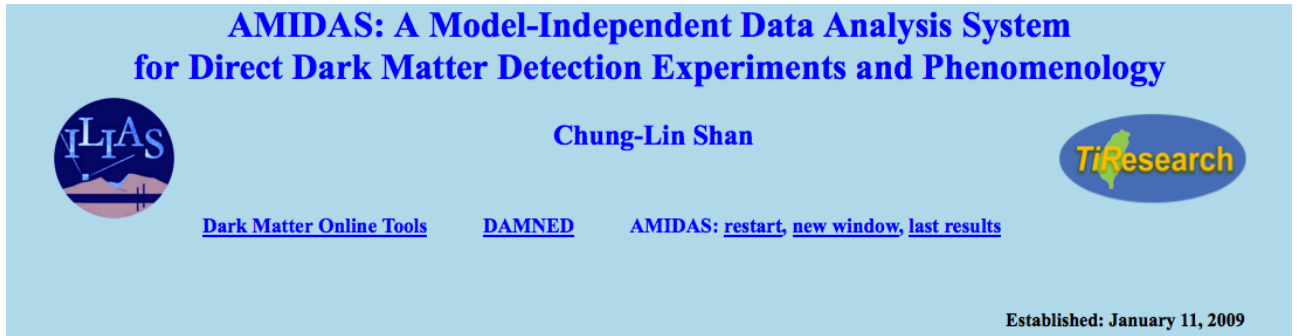
*Kavli Institute for Theoretical Physics China, Chinese Academy of Sciences
No. 55, Zhong Guan Cun East Street, Beijing 100190, China*

E-mail: clshan@phys.nthu.edu.tw

Abstract

In this paper, we give a detailed user's guide to the **AMIDAS** (A Model-Independent Data Analysis System) (package and) website, which is developed for online simulations and data analyses for direct Dark Matter detection experiments and phenomenology. Recently, the whole (**AMIDAS** package and website) system has been upgraded to the second phase: **AMIDAS-II**, for including the new developed Bayesian analysis technique.

AMIDAS(-II) has the ability to do full Monte Carlo simulations as well as to analyze (real/pseudo-) data sets either generated by another event generating programs or recorded in direct DM detection experiments. Moreover, **AMIDAS(-II)** package can include several “user-defined” functions into the main code: the (fitting) one-dimensional WIMP velocity distribution function, the nuclear form factors for spin-independent and spin-dependent cross sections, artificial/experimental background spectrum for both of simulation and data analysis procedures, as well as different distribution functions needed in Bayesian analyses.



1 Introduction

Weakly Interacting Massive Particles (WIMPs) χ arising in several extensions of the Standard Model of electroweak interactions are one of the leading candidates for Dark Matter (DM). Currently, direct DM detection experiments based on measuring recoil energy deposited in a low-background underground detector by elastic scattering of ambient WIMPs off target nuclei are one of the most promising methods for understanding DM properties, identifying them among new particles produced (hopefully in the near future) at colliders, as well as studying the (sub)structure of our Galactic halo (for reviews, see Refs. [1, 2, 3]).

Since 2007 we develop a series of new methods for analyzing data, i.e. measured recoil energies, from direct detection experiments as model-independently as possible. Up to now we could in principle reconstruct the (moments of the) one-dimensional velocity distribution function of halo WIMPs [4, 5], as well as determine the WIMP mass [6] and (ratios between) different WIMP couplings/cross sections on nucleons [7, 8].

Following the development of these model-independent data analysis procedures, we combined the code for our simulations to a compact system: **AMIDAS** (A Model-Independent Data Analysis System). Meanwhile, we modified the code to be able to analyze external data sets either generated by another event generating programs or recorded in direct DM detection experiments. Under the collaboration with the ILIAS Project [9] and the Dark Matter Network Exclusion Diagram (DAMNED, a.k.a. Dark Matter Online Tools) [10], an online system has also been established in January 2009 [11, 12], in order to offer an easier, more convenient and user-friendly environment for simulations and data analyses in (in)direct Dark Matter detection experiments and phenomenology.

In the first phase of the **AMIDAS** package and website, the options for target nuclei, for the velocity distribution function of halo WIMPs, as well as for the elastic nuclear form factors for spin-independent (SI) and spin-dependent (SD) WIMP-nucleus interactions are fixed and only some commonly used analytic forms have been defined [13]. Users can not choose different detector materials nor use different WIMP velocity distribution function nor nuclear form factors for their simulations and/or data analyses. In order to offer more flexible use of the WIMP velocity distribution as well as the nuclear form factors, the **AMIDAS** package has been extended to be more user-oriented and able to include *user-uploaded* files for defining their own functional forms in September 2009 [14]¹.

¹Note that, since the **AMIDAS** code has been written in the C programming language, all user-defined functions to be included into the **AMIDAS** package must be given in the syntax of C. On the other hand, for drawing output plots, since the **Gnuplot** package has been used in **AMIDAS**, the uploaded files for drawing e.g. the (expected) spectrum of measured recoil energy, must be given in the syntax of **Gnuplot** [15]. More detailed descriptions will be given in Secs. 3 and 5 as well as in Appendix.

In this paper, we give a detailed user’s guide to the **AMIDAS** (package and) website. In Sec. 2, we list the **AMIDAS** functions and different working (simulation and data analysis) modes for these functions. The use of intrinsically saved element data to set information of users’ favorite target nuclei will be described in detail. In Sec. 3, we describe the meanings (and options) of all input parameters/factors for running Monte Carlo simulations. The preparation of uploaded files for defining the one-dimensional WIMP velocity distribution function, the elastic nuclear form factors for SI and SD WIMP–nucleus cross sections as well as artificial/experimental background spectrum will be particularly described. In Sec. 4, the preparation of the (real/pseudo–) data files and the uploading/analyzing procedure on the **AMIDAS** website will be given in Sec. 4. The new developed Bayesian analysis technique [5] will be talked separately in Sec. 5. We conclude in Sec. 6. Some technical detail in the **AMIDAS** package will be given in Appendix.

2 AMIDAS functions, working modes, and target nuclei

In this section, we list the functions of the **AMIDAS** (package and) website and different working (simulation and data analysis) modes for these functions. The default options for the target nuclei used for different **AMIDAS** functions as well as the use of intrinsically saved element data to set information of users’ favorite target nuclei will also be described.

2.1 AMIDAS functions

Based on our works on the model-independent data analysis methods for extracting properties of Galactic WIMP DM particles [4, 5, 6, 7, 8], **AMIDAS** has so far the following functions:

List of AMIDAS functions

Choose one of the following functions

- ☐ Generation of WIMP signals with/without background events
- ☒ (Bayesian) reconstruction of the one-dimensional velocity distribution function of halo WIMPs ^{new}
- ☐ Determination of the WIMP mass
- ☐ Estimation of the spin-independent (SI) WIMP-nucleon coupling
- ☐ Determinations of ratios between different WIMP-nucleon couplings/cross sections

2.2 Reconstruction modes

Corresponding to each of the above listed **AMIDAS** functions, there are several different reconstruction modes for users to choose.

2.2.1 For the WIMP–nucleus interaction

For generating WIMP signals (with/without background events), one needs to choose either the SI scalar WIMP–nucleus interaction or the SD axial–vector one is dominated:

Generation of WIMP signals with/without background events

Choose the WIMP-nucleus cross section

- ☒ Only the spin-independent (SI) scalar cross section
- ☐ Only the spin-dependent (SD) axial-vector cross section
- ☐ Both of the SI and SD cross sections

2.2.2 For the needed WIMP mass

For (Bayesian) reconstructing the one-dimensional WIMP velocity distribution function $f_1(v)$ and estimating the SI WIMP–nucleon coupling $|f_p|^2$, one needs the WIMP mass m_χ as an input parameter [4, 5, 7]. This information could be obtained either from e.g. collider experiments or from two (other) direct DM detection experiments [6]. For these two cases, AMIDAS has three options of the input WIMP mass:

Reconstruction of the one-dimensional velocity distribution function of halo WIMPs

Choose the reconstruction mode

- ☐ Only with an input WIMP mass from other/collider experiments
- ☒ Only with a reconstructed WIMP mass from other direct detection experiments
- ☐ With both of them

2.2.3 For the reconstruction of the WIMP mass

In addition, AMIDAS offers two modes for reconstructing the (required) WIMP mass m_χ [6]:

Determination of the WIMP mass

Choose the determination mode

- ☒ Only the combined result from the estimators for different moments
- ☐ Both the combined result from and each of the estimators for different moments

2.2.4 For the reconstruction of the ratio between two SD WIMP–nucleon couplings

Based on the assumption about the (dominated) WIMP–nucleus interaction, we obtained two expressions for reconstructing the ratio between two SD WIMP–nucleon couplings a_n/a_p [8]:

Determinations of ratios of different WIMP-nucleon couplings/cross sections

Choose the determination mode for a_n / a_p

- ☐ Only under the assumption that the SD WIMP-nucleus cross section dominates
- ☐ Only for a general combination of the SI and SD cross sections
- ☒ Both of them
- ☐ None of them - only determine $\sigma_{\chi(p,n)}^{SD} / \sigma_{\chi p}^{SI}$

2.2.5 For the reconstructions of the ratios between SD and SI WIMP–nucleon cross sections

Similarly, by using different detector materials with different spin sensitivities with (un–paired) protons or neutrons, we have also two ways for reconstructing the ratios between the SD and SI WIMP–nucleon cross sections $\sigma_{\chi(p,n)}^{\text{SD}}/\sigma_{\chi p}^{\text{SI}}$ [8]:

Choose the determination mode for $\sigma_{\chi(p,n)}^{\text{SD}}/\sigma_{\chi p}^{\text{SI}}$

- ☐ None of them - only determine a_n/a_p
- ☐ With three targets (two of them are SD sensitive)
- ☐ With two targets (one of them is SD sensitive)
- ☒ Both of them

2.3 Target nuclei

Four frequently used detector materials: ^{28}Si , ^{76}Ge , ^{40}Ar and ^{136}Xe are given as default options of target nuclei for simulations and data analyses by using AMIDAS package. Meanwhile, since September 2009 it is achieved to let users set the target nuclei freely, with (corresponding) atomic number Z and atomic mass number A as well as the total nuclear spin J and the expectation values of the proton and neutron group spins $\langle S_{p,n} \rangle$ for each target nucleus.

2.3.1 Only one required target nucleus

For generating WIMP signals as well as (Bayesian) reconstructing the one–dimensional WIMP velocity distribution function and estimating the SI WIMP–nucleon coupling with an *input* WIMP mass (from other/collider experiments), only one target nucleus is required and users can choose it from the four default options or type the element symbol of the target nucleus, and give the corresponding atomic information: Z , A , J , $\langle S_{p,n} \rangle$ by hand on the website directly:

Choose the target nucleus for generating events

- ☐ ^{28}Si
- ☐ ^{76}Ge
- ☐ ^{40}Ar
- ☐ ^{136}Xe
- ☒ User-defined target: , $Z =$, $A =$, $J =$ / 2, $\langle S_p \rangle =$, $\langle S_n \rangle =$

Note that, firstly, only for the AMIDAS function of generating WIMP signals with either only the SD axial-vector cross section or both of the SI and SD cross sections, the typing cells for the total nuclear spin J and the expectation values of the proton and neutron group spins $\langle S_{p,n} \rangle$ for each target nucleus will appear, in order to remind users that these data are required². Secondly, the given value for the total nuclear spin J is only the *numerator* of the actually value. This means that, for instance, for the use of ^{127}I one needs only give “5” in the typing cell between “ $J =$ ” and “/2” for assigning its J value of 5/2 (see Table 1).

²The same panel used here without the input cells of the J and $\langle S_{p,n} \rangle$ values is also used for the third required target nucleus for reconstructing the ratio between two SD WIMP–nucleon couplings under the consideration of a general combination of the SI and SD WIMP–nucleus cross sections.

Isotope	Z	A	J	$\langle S_p \rangle$	$\langle S_n \rangle$
${}^7\text{Li}$	3	7	3/2	0.497	0.004
${}^{17}\text{O}$	8	17	5/2	0.0	0.495
${}^{19}\text{F}$	9	19	1/2	0.441	-0.109
${}^{23}\text{Na}$	11	23	3/2	0.248	0.020
${}^{27}\text{Al}$	13	27	5/2	0.343	0.030
${}^{29}\text{Si}$	14	29	1/2	-0.002	0.130
${}^{35}\text{Cl}$	17	35	3/2	-0.059	-0.011
${}^{37}\text{Cl}$	17	37	3/2	-0.058	0.050
${}^{39}\text{K}$	19	39	3/2	-0.180	0.050
${}^{73}\text{Ge}$	32	73	9/2	0.030	0.378
${}^{93}\text{Nb}$	41	93	9/2	0.460	0.080
${}^{125}\text{Te}$	52	125	1/2	0.001	0.287
${}^{127}\text{I}$	53	127	5/2	0.309	0.075
${}^{129}\text{Xe}$	54	129	1/2	0.028	0.359
${}^{131}\text{Xe}$	54	131	3/2	-0.009	-0.227
${}^{133}\text{Cs}$	55	133	7/2	-0.370	0.003
${}^{183}\text{W}$	74	183	1/2	0.0	-0.031

Table 1: List of the intrinsically defined nuclear spin data in the **AMIDAS-II** package. More details can be found in e.g. Refs. [1, 16, 17, 18].

2.3.2 Intrinsically defined nuclear spin data

In the upgraded **AMIDAS-II** package, we have further saved the basic atomic data for all 118 elements as well as the nuclear spin data (J and $\langle S_{p,n} \rangle$ values) for 17 frequently used detector nuclei (listed in Table 1). Hence, for setting user-required target nuclei for e.g. generating WIMP-nucleus scattering events, users need only simply to give either the atomic symbol or the atomic number Z (since these two items are one-to-one corresponding). Then all other atomic and nuclear spin information about this chosen nucleus will be shown automatically in the following typing cells.

Moreover, according to the users' assumption about the WIMP-nucleus interaction (SI, SD or both), the **AMIDAS** website will choose automatically the isotope of the chosen nucleus without or with spin sensitivity with the highest natural abundance. For example, when one types for either the defined target “Xe” or the atomic number Z “54”, in the cell of the atomic mass number A “132” or “129” will appear under the choice of the SI or SD (or both) WIMP-nucleus interaction(s). For the latter case, once the nuclear spin data of the chosen nucleus is given intrinsically in the **AMIDAS-II** package (listed in Table 1), the rest required information about the J and $\langle S_{p,n} \rangle$ values will also appear directly:

Generation of WIMP signals with/without background events

Choose the WIMP-nucleus cross section

- ☐ Only the spin-independent (SI) scalar cross section
- ☐ Only the spin-dependent (SD) axial-vector cross section
- ☐ Both of the SI and SD cross sections

Choose the target nucleus for generating events

- ☐ ^{28}Si
- ☐ ^{76}Ge
- ☐ ^{40}Ar
- ☐ ^{136}Xe
- ☒ User-defined target: , $Z =$, $A =$

Generation of WIMP signals with/without background events

Choose the WIMP-nucleus cross section

- ☐ Only the spin-independent (SI) scalar cross section
- ☒ Only the spin-dependent (SD) axial-vector cross section
- ☐ Both of the SI and SD cross sections

Choose the target nucleus for generating events

- ☐ ^{28}Si
- ☐ ^{76}Ge
- ☐ ^{40}Ar
- ☐ ^{136}Xe
- ☒ User-defined target: , $Z =$, $A =$, $J =$ / 2, $\langle S_p \rangle =$, $\langle S_n \rangle =$

Note that, firstly, while the one-to-one corresponding atomic symbol and atomic number Z appear simultaneously, one could modify the atomic mass number A by hand:

Choose the target nucleus for generating events

- ☐ ^{28}Si
- ☐ ^{76}Ge
- ☐ ^{40}Ar
- ☐ ^{136}Xe
- ☒ User-defined target: , $Z =$, $A =$

Choose the target nucleus for generating events

☐ ^{28}Si
☐ ^{76}Ge
☐ ^{40}Ar
☐ ^{136}Xe
☒ User-defined target: Xe , $Z = 54$, $A = 131$, $J = 3$ / 2, $\langle S_p \rangle = -0.009$, $\langle S_n \rangle = -0.227$

Secondly, for one target nucleus, the expected proton and neutron group spins are sometimes model-dependent: e.g. for ^{129}Xe and ^{131}Xe nuclei, $\langle S_p \rangle_{^{129}\text{Xe}} = -0.002$, $\langle S_n \rangle_{^{129}\text{Xe}} = 0.273$ and $\langle S_p \rangle_{^{131}\text{Xe}} = -0.0007$, $\langle S_n \rangle_{^{131}\text{Xe}} = -0.125$ are also often used [19, 20, 21, 22]. These model-dependent values are however not saved in the AMIDAS package yet and thus have to be given by hand:

Choose the target nucleus for generating events

☐ ^{28}Si
☐ ^{76}Ge
☐ ^{40}Ar
☐ ^{136}Xe
☒ User-defined target: Xe , $Z = 54$, $A = 129$, $J = 1$ / 2, $\langle S_p \rangle = -0.002$, $\langle S_n \rangle = 0.273$

Choose the target nucleus for generating events

☐ ^{28}Si
☐ ^{76}Ge
☐ ^{40}Ar
☐ ^{136}Xe
☒ User-defined target: Xe , $Z = 54$, $A = 131$, $J = 3$ / 2, $\langle S_p \rangle = -0.0007$, $\langle S_n \rangle = -0.125$

Moreover, once the J and $\langle S_{p,n} \rangle$ values of the chosen nucleus are *not* defined intrinsically in the AMIDAS-II package, “undefined” would be shown in these cells:

Choose the target nucleus for generating events

☐ ^{28}Si
☐ ^{76}Ge
☐ ^{40}Ar
☐ ^{136}Xe
☒ User-defined target: He , $Z = 2$, $A = 3$, $J = \text{undefin}$ / 2, $\langle S_p \rangle = \text{undefined}$, $\langle S_n \rangle = \text{undefined}$

2.3.3 For the determination of the WIMP mass

For determining the WIMP mass m_χ , two combinations have been considered and programmed in the AMIDAS package [6]:

☐ $^{28}\text{Si} + ^{76}\text{Ge}$

☐ $^{40}\text{Ar} + ^{136}\text{Xe}$

☐ User-defined lighter target: , Z = , A =

heavier target: , Z = , A =

2.3.4 For the determination of the ratio between two SD WIMP–nucleon couplings

Choose the combination of target nuclei for reconstructing a_n / a_p and $\sigma_{\chi(p,n)}^{SD} / \sigma_{\chi p}^{SI}$

☐ $^{73}\text{Ge} + ^{37}\text{Cl}$
☒ $^{19}\text{F} + ^{127}\text{I}$
☐ User-defined first target: , $Z =$, $A =$, $J =$ / 2, $\langle S_p \rangle =$, $\langle S_n \rangle =$
 second target: , $Z =$, $A =$, $J =$ / 2, $\langle S_p \rangle =$, $\langle S_n \rangle =$

2.3.5 For the determination of the ratio between SD and SI WIMP–proton cross section

Choose the combination of target nuclei for reconstructing $\sigma_{\chi p}^{\text{SD}} / \sigma_{\chi p}^{\text{SI}}$

☒ $^{23}\text{Na} + ^{76}\text{Ge}$
☐ User-defined first target: , $Z =$, $A =$, $J =$ / 2, $\langle S_p \rangle =$, $\langle S_n \rangle =$
 second target: , $Z =$, $A =$

³Nevertheless, the **AMIDAS** website can rearrange the order of the chosen nuclei according to the given atomic mass numbers A .

2.3.6 For the determination of the ratio between SD and SI WIMP–neutron cross section

For determining the ratio between SD and SI WIMP–neutron cross section $\sigma_{\chi n}^{\text{SD}}/\sigma_{\chi p}^{\text{SI}}$, two combinations of target nuclei has been considered and programmed in the **AMIDAS** package [8]:

Choose the combination of target nuclei for reconstructing $\sigma_{\chi n}^{\text{SD}}/\sigma_{\chi p}^{\text{SI}}$

☐ $^{17}\text{O} + ^{76}\text{Ge}$

☒ $^{131}\text{Xe} + ^{76}\text{Ge}$

☐ User-defined first target: , $Z =$, $A =$, $J =$ / 2, $\langle S_p \rangle =$, $\langle S_n \rangle =$

second target: , $Z =$, $A =$

Note here that the *first* nucleus of the user–chosen target combination should be spin–sensitive with a (*very*) *small* or even *negligible* expectation value of the *proton* group spin, whereas the *second* one should be *spin–nonsensitive* (without unpaired protons nor neutrons).

2.4 Data type

The probably most important and useful design of the **AMIDAS** package and website is the ability of *not only* doing simulations with self–generated events based on Monte Carlo method, *but also* analyzing user–uploaded (real/pseudo–) data set(s) either generated by other event generators or recorded in direct DM detection experiments *without* modifying the source code.

2.4.1 Data type

Corresponding to our design of the ability of doing Monte Carlo simulations as well as analyzing (real/pseudo–) data set(s), users have two options for the data type:

Data type

Choose the data type

☒ Simulation (events will be generated by AMIDAS)

☐ Real (pseudo–) data

More detailed descriptions about the preparation of data files and the uploading/analyzing procedure on the **AMIDAS** website will be given in Sec. 4.

2.4.2 Simulation mode

One can run numerical Monte Carlo simulations with all **AMIDAS**(–II) functions (given in Sec. 2.1)⁴. Meanwhile, since the running time of the algorithmic procedure for the reconstruction of the WIMP mass m_χ , needed also for the (Bayesian) reconstruction of the one–dimensional WIMP velocity distribution function $f_1(v)$ and the estimation of the SI WIMP–nucleon coupling $|f_p|^2$, is pretty long, **AMIDAS** also offers users faster theoretical estimations as an alternative option:

⁴Note that, considering the current experimental sensitivity and the required executing time for these simulations, the **AMIDAS** website offers full Monte Carlo simulations with maximal 2,000 experiments and maximal 5,000 events (on average) per one experiment.

Choose the simulation mode

- ☒ Monte Carlo simulation
- ☐ Theoretical estimation

Here, instead of the summations of the (moments of the) measured recoil energies required in our model-independent data analysis procedures [4, 6, 7, 8], numerical integrals over the theoretically predicted recoil spectrum will be used. Note however that, firstly, since for these estimations the statistical fluctuations have *not* been taken into account, these pure theoretically estimated results, especially for cases with only (very) few events, could be (fairly) *different* from results obtained by more realistic Monte Carlo simulations. Secondly, as the alternative option for Monte Carlo simulations with much shorter required executing time, the total event number used for these theoretical estimations is fixed⁵ and the calculations are limited to be done for only a few times. These restrictions could sometimes cause unexpected zigzags on the result curves.

Note also here that, once residue background events are taken into account, users have to choose “Monte Carlo simulation”!

2.4.3 Output plots

In our work on the reconstruction of the WIMP mass, we discussed also the statistical fluctuation of the reconstructed m_χ in the simulated experiments [6]. Users can thus choose whether and for what WIMP mass they need the plot of the statistical fluctuation of the reconstructed WIMP mass:

Choose the output plots

- ☒ Only the reconstructed WIMP mass
- ☐ Also the statistical fluctuation for the WIMP mass of GeV

2.4.4 Simulation mode for a_n/a_p and $\sigma_{\chi(p,n)}^{\text{SD}}/\sigma_{\chi p}^{\text{SI}}$

In our works on the determinations of ratios between different WIMP–nucleon couplings/cross sections, we considered two cases separately:

Choose the simulation mode for a_n/a_p and $\sigma_{\chi(p,n)}^{\text{SD}}/\sigma_{\chi p}^{\text{SI}}$

- ☒ Fix m_χ
- ☐ Fix a_n/a_p

2.4.5 Background events

After the development of our model-independent methods for reconstructing different WIMP properties, we worked also on effects of residue background events in analyzed data sets [23, 24, 25, 26]. Hence, users have the option to do simulations with or without an intrinsically or user-defined (uploaded) background spectrum:

⁵In contrast and more realistically, the actual number(s) of generated WIMP signals (and background events) for each Monte Carlo simulated experiment is Poisson-distributed around the expected value(s) set by users.

Take into account background events

- ☐ Yes
☒ No

More detailed descriptions about intrinsically defined background spectra as well as the file preparation of users' own artificial/experimental background spectrum will be given in Sec. 3.5.

2.4.6 Generated events

As an extra service, users can receive *all* WIMP–signal (and background) events generated by AMIDAS in separate text files for different target nuclei:

Output generated events

- ☐ Yes
☒ No

Note here that, for the determination of the WIMP mass, the estimation of the SI WIMP–nucleon coupling as well as the determinations of ratios between different WIMP–nucleon couplings/cross sections with several different input WIMP masses, the required value of the input WIMP mass has to be chosen:

Output generated events

- ☐ Yes, for the WIMP mass of GeV
☒ No

Similarly, for the determinations of ratios between different WIMP–nucleon couplings/cross sections with several different input SD WIMP–nucleon coupling ratios, users have to choose the required a_n/a_p value:

Output generated events

- ☐ Yes, for the a_n / a_p ratio of
☒ No

3 Running simulations

In this section, we describe the meanings (and options) of all input parameters/factors needed in principle *only* for predicting the recoil and background spectra for generating WIMP signal and residue background events and for drawing the output plots. Some commonly used and/or standard AMIDAS simulation/analysis values used in our works presented in Refs. [4, 5, 6, 7, 8, 23, 24, 25, 26] have been given as default, but users can choose or modify all these parameters/options by hand.

Moreover, the preparation of uploaded files for defining the one-dimensional WIMP velocity distribution function, the elastic nuclear form factors for SI and SD WIMP–nucleus cross sections as well as artificial/experimental background spectrum will be particularly described.

3.1 WIMP properties

The following information on WIMP properties is required for predicting the recoil spectrum and/or analyzing user-uploaded data:

Symbol	Meaning	Remarks
m_χ	The input WIMP mass	GeV/c^2
$\sigma(m_\chi)$	An overall uncertainty on the input WIMP mass	[0, 1]
$\sigma_{\chi p}^{\text{SI}}$	The SI WIMP–proton cross section	pb
f_n/f_p	The ratio of the SI WIMP coupling on neutrons to that on protons	
a_p	The SD WIMP–proton coupling	
a_n/a_p	The ratio of the SD WIMP coupling on neutrons to that on protons	

Simulation setup

WIMP properties

$m_\chi (\text{GeV}/c^2)$	$\sigma(m_\chi)$	$\sigma_{\chi p}^{\text{SI}} (\text{pb})$	f_n/f_p	a_p	a_n/a_p
100	0.05	1e-9	1.0	0.1	0.7

* Move the cursor onto a symbol for checking its definition.

Note that, firstly, *not* all of these items are needed for every **AMIDAS** function. The website will give the needed items automatically according to users earlier options. Secondly, in case that any required datum is missed, this omission will be detected automatically after the submission and users will be reminded of that with a *red* block around the table. For this case *all data* in this setup table will be *reset* to the default values and should therefore be checked and modified once again to the users' own setup. Remind also that, the overall uncertainty on the input WIMP mass should be between 0 and 1. On the other hand, users can hover the curser onto an item symbol in the setup table for checking its definition.

3.2 Astronomical setup

AMIDAS requires also information on the following astronomical parameters for predicting/fitting the velocity distribution function of halo WIMPs as well as for estimating the SI WIMP–nucleon coupling:

Symbol	Meaning	Remarks
ρ_0	The WIMP density near the Earth	$\text{GeV}/c^2/\text{cm}^3$
v_0	The Sun's orbital speed around the Galactic center	km/s
v_{max}	The maximal cut-off on the 1-D WIMP velocity distribution function	km/s
t_p	The date on which the Earth's velocity relative to the WIMP halo is maximal	day
t_{expt}	The (middle of) the running date of the experiment	day

Astronomical setup

ρ_0 (GeV/c ² /cm ³)	v_0 (km/s)	v_{\max} (km/s)	t_p (day)	t_{expt} (day)
0.3	220	700	152.5	243.75

* Move the cursor onto a symbol for checking its definition.

Remind that, in case that any required datum is missed, this omission will be detected automatically after the submission and users will be reminded of that with a *red* block around the table. For this case *all data* in this setup table will be *reset* to the default values and should therefore be checked and modified once again to the users' own setup. On the other hand, users can hover the cursor onto an item symbol in the setup table for checking its definition.

3.3 Velocity distribution function of halo WIMPs

For predicting the elastic WIMP–nucleus scattering spectrum in order to generate WIMP–induced signal events, one needs crucially the one–dimensional WIMP velocity distribution function $f_1(v)$. In the AMIDAS–II package, we define intrinsically three most commonly used theoretical distributions. Meanwhile, users also have the option to define (by uploading/typing) an analytic form of their favorite one–dimensional velocity distribution:

Choose the velocity distribution function

☐ Simple Maxwellian [analytic form](#)
☐ Modified Maxwellian with $k =$ [analytic form](#)
☒ Shifted Maxwellian [analytic form](#)
☐ User-defined distribution function

Upload files: 未選擇檔案。

未選擇檔案。 (for drawing output plots)

Type definitions:

```
double Intflv_v_user(double mchi, int A, double QQ, double tt)
{
  return
    (2.0 / sqrt(M_PI) / (v_0 * v_U)) *
    exp(-alpha(mchi, A) * alpha(mchi, A) * QQ /
      (v_0 * v_U + (v_0 * v_U) * alpha(mchi, A) * alpha(mchi, A) * QQ))
}

M_PI = 3.141593
c = 1.0
m_U = 1e6 / (c * c)
```

(for drawing output plots)

Download: [sample for a user-defined velocity distribution, for drawing output plots](#)

Note here that, firstly, for checking the analytic forms of the intrinsically defined velocity distributions, users can hover the cursor onto “[analytic form](#)”; users can also click the “[analytic form](#)” to open a new webpage with more detailed information and useful references. Secondly, as reminded on the website, the second uploaded file/typing area is *only* for drawing output plot(s) of the generated WIMP–signal (and background) events and, as a comparison to, the (Bayesian) reconstructed one–dimensional WIMP velocity distribution function. Moreover, consider the normal height of a browser window, we shrink the typing areas as default. However, once users click one of the two typing areas, the clicked one will extend to show the full content; the extended typing area(s) will shrink automatically after users click one of the three default velocity distribution functions.

Choose the velocity distribution function

- ☐ Simple Maxwellian [analytic form](#)
- ☐ Modified Maxwellian with $k =$ [analytic form](#)
- ☐ Shifted Maxwellian [analytic form](#)
- ☒ User-defined distribution function

Upload files:

[浏览...](#)

未選擇檔案

[浏览...](#)

未選擇檔案

(for drawing output plots)

Type definitions:

```
double Intflv_v_user(double m_chi, int A, double QQ, double tt)
{
    return
    (2.0 / sqrt(M_PI) / (v_0 * v_U)) *
    exp(-alpha(m_chi, A) * alpha(m_chi, A) * QQ /
    ( (v_0 * v_U) * (v_0 * v_U) ) );
}
```

```
M_PI = 3.141593
c = 1.0
m_U = 1e6 / (c * c)
v_U = c / (2.9979246 * 1e5)
m_N = (0.938272 * m_U) * AX * 0.99
m_rN = (m_chi * m_U) * m_N / (m_chi * m_U + m_N)
alpha = sqrt(m_N / (2.0 * m_rN * m_rN))
flv_user(x) \
= (4.0 / sqrt(M_PI)) * \
((x * x) / (v_0 * v_0 + v_U)) * \
exp(-(x * x) / (v_0 * v_U)) \
Intflv_v_user(x) \
= (2.0 / sqrt(M_PI) / (v_0 * v_U)) * \
exp(-alpha * alpha * x / ( (v_0 * v_U) * (v_0 * v_U) ) )
```

(for drawing output plots)

Download: [sample for a user-defined velocity distribution, for drawing output plots](#)

In this subsection, we give first the definitions of the three default velocity distribution functions in the **AMIDAS-II** package. Then we will describe how to (modify these definitions to) define user's own velocity distribution.

3.3.1 Default one-dimensional WIMP velocity distribution functions

So far users have three options for the one-dimensional WIMP velocity distribution function defined intrinsically in the **AMIDAS-II** package [4, 5]:

1. the simple Maxwellian velocity distribution function [1]

$$f_{1,\text{Gau}}(v) = \frac{4}{\sqrt{\pi}} \left(\frac{v^2}{v_0^3} \right) e^{-v^2/v_0^2}; \quad (1)$$

2. the modified Maxwellian velocity distribution function [27, 28, 29, 30]

$$f_{1,\text{Gau},k}(v) = \frac{v^2}{N_{f,k}} \left(e^{-v^2/kv_0^2} - e^{-v_{\text{max}}^2/kv_0^2} \right)^k, \quad (\text{for } v \leq v_{\text{max}}), \quad (2)$$

for $k = 1, 2, 3, 4$, where $N_{f,k}$ is the normalization constant depending on the value of the power index k ;

3. the shifted Maxwellian velocity distribution function [1, 31]

$$f_{1,\text{sh}}(v) = \frac{1}{\sqrt{\pi}} \left(\frac{v}{v_0 v_e} \right) \left[e^{-(v-v_e)^2/v_0^2} - e^{-(v+v_e)^2/v_0^2} \right], \quad (3)$$

where the time-dependent Earth’s velocity in the Galactic frame is given as

$$v_e(t) = v_0 \left[1.05 + 0.07 \cos \left(\frac{2\pi(t - t_p)}{1 \text{ yr}} \right) \right]. \quad (4)$$

3.3.2 User-defining the one-dimensional WIMP velocity distribution function

For defining the one-dimensional WIMP velocity distribution (for generating WIMP signals), users should in practice give the *integral* over the velocity distribution function:

$$\int_{v_{\min}=\alpha\sqrt{Q}}^{v_{\max}} \left[\frac{f_1(v, t)}{v} \right] dv \quad (5)$$

with the name of “Intf1v_v_user”. Note here that the lower limit of the integral, the minimal incoming velocity of incident WIMPs that can deposit the energy Q in the detector, v_{\min} , should be expressed as a function of the energy Q through $v_{\min} = \alpha\sqrt{Q}$; the upper limit of the integral, v_{\max} , can be set as one of the required astronomical parameters (see Sec. 3.2). Meanwhile, since the transformation constant

$$\alpha \equiv \sqrt{\frac{m_N}{2m_{r,N}^2}} \quad (6)$$

is defined with the reduced mass of the WIMP mass m_χ and that of the target nucleus m_N

$$m_{r,N} \equiv \frac{m_\chi m_N}{m_\chi + m_N}, \quad (7)$$

and m_N is given as a function of the atomic mass number A in the **AMIDAS** package, the expression of the integral over the WIMP velocity distribution should thus be a function of m_χ and A . Finally, in more general cases, the (integral over the) velocity distribution function should also be a function of time t .

As an example, we give here the **AMIDAS-II** code for the integral over the simple Maxwellian velocity distribution function. The codes for the (integrals over the) other intrinsically defined one-dimensional WIMP velocity distribution functions will be given in Appendix B.1 for users’ references.

Code 1: Integral over the simple Maxwellian velocity distribution $\int_{\alpha\sqrt{Q}}^{\infty} [f_{1,\text{Gau}}(v)/v] dv$

```
double Intf1v_v_user(double mchi, int A, double QQ, double tt)
{
    return
        ( 2.0 / sqrt(M_PI) / (v_0 * v_U) ) *
        exp(-alpha(mchi, A) * alpha(mchi, A) * QQ /
            ( (v_0 * v_U) * (v_0 * v_U) ) );
}
```

Note that, firstly, `v_U` and the function `alpha(mchi, A)`, where `mchi` and `A` stand for the WIMP mass m_χ and the atomic mass number of the target nucleus, A , are defined intrinsically in the **AMIDAS** package⁶. Secondly, `v_0`, standing for v_0 , is an input parameter which users can set separately on the website (see Sec. 3.2). Moreover, `QQ` and `tt` stand for the recoil energy Q and time t . Remind also here that, even for a *time-independent* velocity distribution or a numerical

⁶Relevant constants and functions defined in the **AMIDAS** package will be given in Appendix A.1 and A.2.

expression with a fixed experimental running date $t = t_{\text{expt}}$, “double tt” should always be declared as the *forth* parameter of the function `Intf1v_v_user(mchi, A, QQ, tt)`.

On the other hand, an *extra* file is required for defining the input theoretical one-dimensional velocity distribution function *itself* (not the integral over it!), for drawing this input velocity distribution, as a comparison, with the Bayesian reconstructed one together in the output plots. Meanwhile, the *integral* over this input velocity distribution is however needed for drawing the theoretical WIMP scattering spectrum, with the (binned) recorded WIMP-signals (and background) events together in the output plots. Since the **Gnuplot** software has been adopted in the **AMIDAS** package for drawing output plots, the (integral over the) velocity distribution function defined in this file must be written in the syntax of **Gnuplot** with the name of “`f1v_user(x)`” or “`Intf1v_v_user(x)`”. Below we give our definitions for the (integral over the) simple Maxwellian velocity distribution as examples. The definitions for the (integral over the) other intrinsically defined one-dimensional WIMP velocity distribution function for **Gnuplot** are given in Appendix B.1 for users’ references.

Code 2: Simple Maxwellian velocity distribution $f_{1,\text{Gau}}(v)$ (for Gnuplot)

```
M_PI = 3.141593

f1v_user(x)
= (4.0 / sqrt(M_PI)) * \
  ( (x * x) / (v_0 * v_0 * v_0) ) * \
  exp(-(x * x) / (v_0 * v_0))
```

Code 3: Integral over the simple Maxwellian velocity distribution $\int_{\alpha\sqrt{Q}}^{\infty} [f_{1,\text{Gau}}(v)/v] dv$ (for Gnuplot)

```
c = 1.0
m_U = 1e6 / (c * c)
v_U = c / (2.9979246 * 1e5)

m_N = (0.938272 * m_U) * AX * 0.99
m_rN = (m_chi * m_U) * m_N / (m_chi * m_U + m_N)

alpha = sqrt(m_N / (2.0 * m_rN * m_rN))

Intf1v_v_user(x)
= (2.0 / sqrt(M_PI) / (v_0 * v_U)) * \
  exp(-alpha * alpha * x / ( (v_0 * v_U) * (v_0 * v_U) ))
```

Note here that, firstly, “\” (backslash) must be used in order to let the including of the definition(s) given in this file into the other intrinsic commands correctly. Secondly, three flexible-kept parameters: **AX**, **m_chi** and **v_0** will be read directly from the users’ initial (simulation/data analysis) setup set earlier on the website. The meanings of the variables used here can be found in Appendix A.1.

Remind also that two sample files, one for the **AMIDAS** code and the other one for the **Gnuplot** package, can be downloaded from the **AMIDAS** website.

3.4 Elastic nuclear form factors for the WIMP–nucleus cross sections

For predicting the elastic WIMP–nucleus scattering spectrum in order to generate WIMP-induced signal events as well as reconstructing different WIMP properties: the one-dimensional

velocity distribution function $f_1(v)$, the mass m_χ , the SI coupling on nucleons $|f_p|^2$ and the ratios between different WIMP–nucleon couplings/cross sections a_n/a_p and $\sigma_{\chi(p,n)}^{\text{SD}}/\sigma_{\chi p}^{\text{SI}}$, one needs crucially the elastic nuclear form factor for SI WIMP–nucleus interaction, $F_{\text{SI}}^2(Q)$. In the **AMIDAS** package, we define intrinsically four most commonly used elastic nuclear form factors for the SI cross section. Meanwhile, users also have the option to define (by uploading/typing) an analytic form of their favorite form factor:

Choose the elastic nuclear form factor for the SI WIMP-nucleus cross section

- ☐ Exponential analytic form
- ☒ Woods-Saxon analytic form
- ☐ Woods-Saxon with a modified nuclear radius analytic form
- ☐ Helm analytic form
- ☐ User-defined SI form factor

Upload files: 未选择檔案 + 未选择檔案 + (for drawing output plots)

Type definitions:

```
double FQ_SI_user(int A, double QQ)
{
    return exp(-QQ / Q_0(A));
}

double dFQ_SI_user(int A, double QQ)
{
    c = 1.0
    m_U = 1e6 / (c * c)
    fm_U = c / (0.197327 * 1e6)
}
```

(for drawing output plots)

Download: [sample for a user-defined form factor, for drawing output plots](#)

On the other hand, for generating WIMP–induced signal events (only) with SD WIMP–nucleus interaction as well as reconstructing the ratios between different WIMP–nucleon couplings/cross sections a_n/a_p and $\sigma_{\chi(p,n)}^{\text{SD}}/\sigma_{\chi p}^{\text{SI}}$, one needs crucially the elastic nuclear form factor for SD WIMP–nucleus interaction, $F_{\text{SD}}^2(Q)$. In the **AMIDAS** package, so far we define intrinsically only one elastic nuclear form factor for the SD cross section. As usual, users also have the option to define (by uploading/typing) an analytic form of their favorite form factor:

Choose the elastic nuclear form factor for the SD WIMP-nucleus cross section

- ☒ Thin-shell analytic form
- ☐ User-defined SD form factor

Upload files: 未选择檔案 + 未选择檔案 + (for drawing output plots)

Type definitions:

```
double FQ_SD_user(int A, double QQ)
{
    if (QQ == 0.0)
    {
        return 1.0;
    }
    c = 1.0
    m_U = 1e6 / (c * c)
    fm_U = c / (0.197327 * 1e6)
}
```

(for drawing output plots)

Download: [sample for a user-defined form factor, for drawing output plots](#)

Note here that, firstly, for checking the analytic forms of the intrinsically defined SI and SD elastic nuclear form factors, users can hover the cursor onto “**analytic form**”; users can also click the “**analytic form**” to open a new webpage with more detailed information and useful references. Secondly, as reminded on the website, the second uploaded file/typing area is *only* for drawing output plot(s) of the generated WIMP–signal (and background) events.

In this subsection, we give first the definitions of the four default elastic nuclear form factors for the SI WIMP–nucleus cross section and the unique one for the SD cross section in the **AMIDAS** package. Then we will describe how to (modify these definitions to) define user’s own form factor.

3.4.1 Default SI elastic nuclear form factors

So far users have four options for the SI elastic nuclear form factor defined intrinsically in the AMIDAS package [4, 32]:

1. the exponential form factor [33, 34, 1]

$$F_{\text{ex}}^2(Q) = e^{-Q/Q_0}, \quad (8)$$

where

$$Q_0 = \frac{1.5}{m_N R_0^2} \quad (9)$$

is the nuclear coherence energy and

$$R_0 = \left[0.3 + 0.91 \left(\frac{m_N}{\text{GeV}} \right)^{1/3} \right] \text{ fm} \quad (10)$$

is the radius of the nucleus;

2. the Woods–Saxon form factor [35, 1, 31]

$$F_{\text{WS}}^2(Q) = \left[\frac{3j_1(qR_1)}{qR_1} \right]^2 e^{-(qs)^2}, \quad (11)$$

where $j_1(x)$ is a spherical Bessel function,

$$q = \sqrt{2m_N Q} \quad (12)$$

is the transferred 3-momentum,

$$R_1 = \sqrt{R_A^2 - 5s^2} \quad (13)$$

is the effective nuclear radius with

$$R_A \simeq 1.2 A^{1/3} \text{ fm}, \quad (14)$$

and

$$s \simeq 1 \text{ fm} \quad (15)$$

is the nuclear skin thickness, A is the atomic mass number of the nucleus;

3. the Woods–Saxon form factor with a modified nuclear radius [36, 31], in which

$$R_A \simeq (1.15 A^{1/3} + 0.39) \text{ fm}; \quad (16)$$

4. the Helm form factor [37, 31]

$$R_1 = \sqrt{R_A^2 + \left(\frac{7}{3}\right)\pi^2 r_0^2 - 5s^2} \quad (17)$$

is the effective nuclear radius with

$$R_A \simeq (1.23 A^{1/3} - 0.6) \text{ fm} , \quad (18)$$

$$r_0 \simeq 0.52 \text{ fm} , \quad (19)$$

and

$$s \simeq 0.9 \text{ fm} \quad (20)$$

is the nuclear skin thickness.

3.4.2 Default SD elastic nuclear form factor

Comparing to the SI case, the nuclear form factor for the SD WIMP–nucleus cross section is more complicated and target–dependent, due to its dependence on the SD WIMP–nucleon couplings as well as on the individual spin structure of target nuclei. Hence, so far in the **AMIDAS** package we define only one general analytic form for the SD elastic nuclear form factor:

1. the thin–shell form factor [31, 38]

$$F_{\text{TS}}^2(Q) = \begin{cases} j_0^2(qR_1), & \text{for } qR_1 \leq 2.55 \text{ or } qR_1 \geq 4.5, \\ \text{const.} \simeq 0.047, & \text{for } 2.55 \leq qR_1 \leq 4.5, \end{cases} \quad (21)$$

where $j_0(x)$ is a spherical Bessel function.

3.4.3 User–defining the elastic nuclear form factor

For defining users’ favorite nuclear form factors, *not only* the definitions of the *squared* form factor $F^2(Q)$ *but also* those of the derivatives of $F^2(Q)$ with respect to the energy Q :

$$\frac{dF^2(Q)}{dQ} = 2F(Q) \left[\frac{dF(Q)}{dQ} \right] \quad (22)$$

(not $F(Q)$ itself nor $dF(Q)/dQ$) must be given *together* in *one* file with (however) the names of “FQ_SI_user” (“FQ_SD_user”) and “dFQdQ_SI_user” (“dFQdQ_SD_user”).

As an example, we give here the **AMIDAS** code for the (derivative of the) exponential form factor. The codes for the (derivatives of the) other intrinsically defined elastic nuclear form factors will be given in Appendix B.2 for users’ references.

Code 4: Squared exponential form factor $F_{\text{ex}}^2(A, Q)$

```
double FQ_SI_user(int A, double QQ)
{
    return exp(-QQ / Q_0(A));
}
```

Code 5: Derivative of the squared exponential form factor $dF_{\text{ex}}^2(A, Q)/dQ$

```
double dFQdQ_SI_user(int A, double QQ)
{
    return -(1.0 / Q_0(A)) * FQ_SI_user(A, QQ);
}
```

Note that the function $Q_0(A)$ is defined intrinsically in the **AMIDAS** package (see Appendix B.2).

On the other hand, *two extra* files are required for defining the elastic nuclear form factors themselves (*without* their derivatives!) for the SI and/or SD WIMP–nucleus cross section(s) *separately*, for drawing the theoretical WIMP scattering spectrum, as a comparison, with the (binned) recorded WIMP–signals (and background) events together in the output plots. Note that the form factors defined in these two files must be written in the syntax of **Gnuplot** with the name of “**FQ_SI_user(x)**” or “**FQ_SD_user(x)**”. Below we give our definition for the exponential form factor as an example. The definitions for the other intrinsically defined elastic nuclear form factors are given in Appendix B.2 for users’ references.

Code 6: Squared exponential form factor $F_{\text{ex}}^2(A, Q)$ (for Gnuplot)

```
c      = 1.0
m_U    = 1e6 / (c * c)
fm_U   = c / (0.197327 * 1e6)

m_N    = (0.938272 * m_U) * AX * 0.99
m_rN   = (m_chi * m_U) * m_N / (m_chi * m_U + m_N)

alpha  = sqrt(m_N / (2.0 * m_rN * m_rN))

R_0    = ( 0.3 + 0.91 * (m_N / m_U) ** (1.0 / 3.0) ) * fm_U
Q_0    = 1.5 / (m_N * R_0 * R_0)

FQ_SI_user(x)  \
= exp(-x / Q_0)
```

Remind that, firstly, “\” (backslash) must be used in order to let the including of the definition given in this file into the other intrinsic commands correctly. Secondly, two flexible-kept parameters: **AX** and **m_chi** will be read directly from the users’ initial (simulation/data analysis) setup set earlier on the website. The meanings of the variables used here can be found in Appendix A.1.

Remind also that two sample files, one for the **AMIDAS** code and the other one for the **Gnuplot** package, can be downloaded from the **AMIDAS** website.

3.5 Background spectrum

As an extra consideration, users have the opportunity to take into account some unrejected background events in numerical simulations with the **AMIDAS-II** package.

So far in the **AMIDAS-II** package, we defined intrinsically three analytic forms for the artificial background spectrum. Meanwhile, users also have the option to define (by uploading/typing) an analytic form of their favorite theoretical or experimental background spectrum:

Choose the background spectrum

☐ Constant [analytic form](#)
☒ Exponential [analytic form](#)
☐ Gaussian-excess with a central energy of keV and a width of keV [analytic form](#)
☐ User-defined background spectrum

Upload files: 未選擇檔案

未選擇檔案 (for drawing output plots)

Type definitions:

```

float dRdQ_bg_mu_user = 50.0;
float dRdQ_bg_sigma_mu_user = 2.0;

float dRdQ_bg_ratio_Gau_user = 1.0;
float dRdQ_bg_ratio_const = 2.0;

M_PI = 3.141593

dRdQ_bg_mu_user = 50.0
dRdQ_bg_sigma_mu_user = 2.0
dRdQ_bg_ratio_Gau_user = 1.0
  
```

(for drawing output plots)

Download: [sample for a user-defined background spectrum](#), [for drawing output plots](#)

Note that, firstly, for checking the analytic forms of the intrinsically defined background spectra, users can hover the cursor onto “**analytic form**”; users can also click the “**analytic form**” to open a new webpage with more detailed information and useful references. Secondly, as reminded on the website, the second uploaded file/typing area is *only* for drawing output plot(s) of the generated WIMP–signal *and* background events.

In this subsection, we give first the definitions of the three (artificial) background spectra in the AMIDAS-II package. Then we will describe how to (modify these definitions to) define user’s own (theoretical or experimental) background spectrum.

3.5.1 Default background spectra

So far users have three options for the (artificial) background spectrum defined intrinsically in the AMIDAS-II package [23, 39]:

1. constant background spectrum [23]

$$\left(\frac{dR}{dQ}\right)_{\text{bg,const}} = 1; \quad (23)$$

2. exponential background spectrum [23]

$$\left(\frac{dR}{dQ}\right)_{\text{bg,ex}} = \exp\left(-\frac{Q/\text{keV}}{A^{0.6}}\right), \quad (24)$$

where A is the atomic mass number of the target nucleus;

3. Gaussian–excess background spectrum [39]

$$\left(\frac{dR}{dQ}\right)_{\text{bg,Gau}} = \frac{1}{\sqrt{2\pi}(\sigma_{Q,\text{bg}}/\text{keV})} \exp\left[-\frac{(Q - Q_{\text{bg,peak}})^2}{2\sigma_{Q,\text{bg}}^2}\right], \quad (25)$$

where $Q_{\text{bg,peak}}$ and $\sigma_{Q,\text{bg}}$ are the central energy and the width of the Gaussian background excess.

3.5.2 User-defining the background spectrum

For defining users' needed theoretical/experimental background spectrum for generating residue background events, users only have to give the analytic form of the spectrum *itself* as a function of the recoil energy Q (QQ in the AMIDAS code) in a file with the name of "dRdQ_bg_user".

As an example, we give here the AMIDAS-II code for the target-dependent exponential background spectrum. The codes for the other intrinsically defined background spectra will be given in Appendix B.3 for users' references.

Code 7: Target-dependent exponential background spectrum $(dR/dQ)_{\text{bg,ex}}(A, Q)$

```
double Q_0_bg_user(int A)
{
    return pow(A, 0.6);
}

double dRdQ_bg_user(int A, double QQ)
{
    return exp(-QQ / Q_0_bg_user(A));
}
```

Note here that, since we take into account the target (experiment) dependence of (the artificially added) background spectrum, the atomic mass number A is used here as the first function parameter (for detailed discussion see Ref. [23]).

On the other hand, an *extra* file is required for defining the background spectrum *itself* for drawing this extra part (with the theoretical WIMP signal spectrum), as a comparison, with the (binned) recorded WIMP-signals and background events together in the output plots. Note that the integral over this background spectrum must also be given in the same file in order to *normalize* the background spectrum properly according to the user's required background ratio (to the WIMP-induced signals). Remind that the (integral over the) background spectrum defined in this file must be written in the syntax of Gnuplot with the name of "dRdQ_bg_user(x)" and "IntdRdQ_bg_user(x)". Below we give our definition for the (integral over the) target-dependent exponential background spectrum as an example. The definitions for the other intrinsically defined (integral over the) background spectra are given in Appendix B.3 for users' references.

Code 8: Target-dependent exponential background spectrum $(dR/dQ)_{\text{bg,ex}}(A, Q)$ (for Gnuplot)

```
dRdQ_bg_user(x)      \
= exp(-x / AX ** 0.6)
```

Code 9: Integral over the target-dependent exponential background spectrum $\int (dR/dQ)_{\text{bg,ex}}(A, Q) dQ$ (for Gnuplot)

```
IntdRdQ_bg_user(x)    \
-- (AX ** 0.6) * exp(-x / AX ** 0.6)
```

Remind that, firstly, "\" (backslash) must be used in order to let the including of the definitions given in this file into the other intrinsic commands correctly. Secondly, the flexible-kept parameters: AX will be read directly from the users' initial (simulation/data analysis) setup set earlier on the website. The meanings of the variables used here can be found in Appendix A.1.

Remind also that two sample files, one for the AMIDAS code and the other one for the Gnuplot package, can be downloaded from the AMIDAS website.

3.6 Experimental setup

Finally, one needs to set the following experimental information for both numerical (Monte Carlo) simulations and data analyses.

3.6.1 Experimental setup

For generating WIMP signals, running numerical simulations as well as analyzing uploaded data sets, the **AMIDAS** package needs

Symbol	Meaning	Remarks
Q_{\min}	The minimal cut-off energy	keV
Q_{\max}	The maximal cut-off energy	keV
b_1	The width of the first Q -bin	keV
N_{tot}	The expected total event number between Q_{\min} and Q_{\max}	Max. 5,000
B	The number of Q -bin between Q_{\min} and Q_{\max}	[4, 10]
N_{expt}	The number of simulated experiment or uploaded data set	Max. 2,000

Experimental setup

Target	Q_{\min} (keV)	Q_{\max} (keV)	b_1 (keV)	N_{tot} (max. 5,000)
^{76}Ge	0	100	10	500
5	Q -bins (min. 4, max. 10)			
500	simulated experiments (max. 2,000)			

* Move the cursor onto a symbol for checking its definition.

Remind that, firstly, the **AMIDAS** website offers full Monte Carlo simulations with maximal 2,000 experiments and maximal 5,000 events (on average) per one experiment; the number of Q -bin between Q_{\min} and Q_{\max} should be between 4 and 10 bins. Secondly, users can hover the cursor onto an item symbol in the setup table for checking its definition.

3.6.2 Experimental setup for background events

Once users want to take into account background events for their numerical simulations, the following information should be given:

Symbol	Meaning	Remarks
$Q_{\min,\text{bg}}$	The lower bound of the background window	keV
$Q_{\max,\text{bg}}$	The upper bound of the background window	keV
r_{bg}	The ratio of background events in the whole data set	[0, 1]
$N_{\text{tot},\text{bg}}$	The expected number of background events in the background window	Calculated automatically
$N_{\text{tot},\text{sg}}$	The expected number of WIMP signals in the background window	Calculated automatically

Experimental setup for background events

Target	$Q_{\min, \text{bg}}$ (keV)	$Q_{\max, \text{bg}}$ (keV)	r_{bg}	$N_{\text{tot, bg}}$	$N_{\text{tot, sg}}$
^{76}Ge	0	100	0.1	50	450

* Move the cursor onto a symbol for checking its definition.

Remind that, firstly, the ratio of background events in the whole data set should be between 0 and 1. By setting the total event number N_{tot} and the background ratio r_{bg} , the (average) numbers of WIMP–signals and residue background events, $N_{\text{tot,sg}}$ and $N_{\text{tot,bg}}$, will be calculated automatically. Secondly, users can hover the cursor onto an item symbol in the setup table for checking its definition.

3.7 Running simulations

After giving all the required information for the aimed simulation, users have one more chance to check their choices, modify some of them, and then resubmit the whole setup, before they click the “Simulation start” button to start the AMIDAS program.

Simulation starting

Check the above setup

Simulation start

Modify and re-submit

Remind that, in case that any required datum is missed, this omission will be detected automatically after the (re)submission and users will be reminded of that with a *red* block around the options/table. Note that *all data* in a table with missed information will be *reset* to the default values and should therefore be checked and modified once again to the users’ own setup.

Once all the required data have been checked, users have only to click the “Simulation start” button and wait for the simulation results for a few minutes⁷.

3.8 Output results (plots)

Simulation results will be presented in form(s) of plot(s) and occasionally table(s). Considering users’ different needs of plot formats, we offer *four* most commonly used file types: PostScript (PS), Encapsulated PostScript (EPS), Portable Document Format (PDF) and Portable Network Graphics (PNG).

⁷For the case that a simulation takes a long time and thus the result is not shown properly, users can check the following URL manually:

<http://pisrv0.pit.physik.uni-tuebingen.de/darkmatter/amidas/amidas-results.php>.

Results

Generated WIMP recoil spectrum with background events

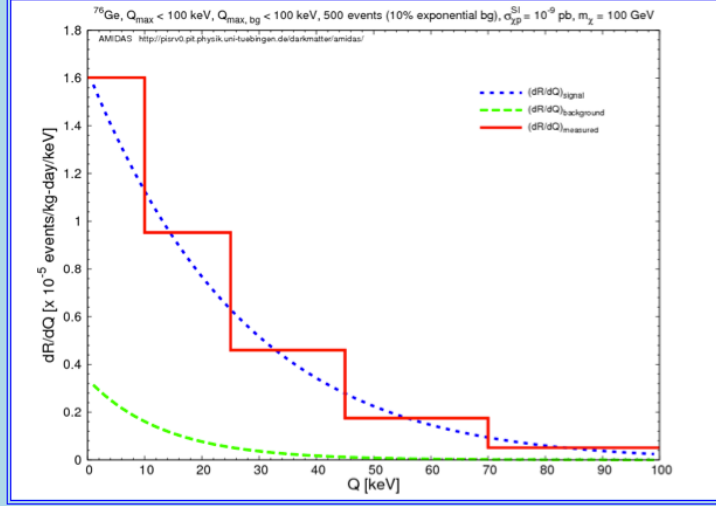
PS: [amidas-140210-112159-01.ps](#)

EPS: [amidas-140210-112159-01.eps](#)

PDF: [amidas-140210-112159-01.pdf](#)

PNG: [amidas-140210-112159-01.png](#)

Plot preview



Plot credit: AMIDAS <http://pirv0.pit.physik.uni-tuebingen.de/darkmatter/amidas/>

The solid red histogram is the recoil spectrum for a ^{76}Ge target and an input WIMP mass of 100 GeV. The dotted blue curve is the elastic WIMP-nucleus scattering spectrum for the shifted Maxwellian velocity distribution of halo WIMPs and the Woods-Saxon nuclear form factor for the SI WIMP interaction, whereas the dashed green curve is the exponential background spectrum given in Eq. (20) of Ref. [5].

Data

TXT: [amidas-140210-112159.txt](#)

TXT: [amidas-140210-112159-data-set-1-Ge76.txt](#)

In order to let users understand the output results more clearly and use them more conveniently, each output plot or table will be accompanied with a short description.

On the other hand, for users' need of self-producing results for different kinds of presentations, the original TXT file(s) of the simulation results with users' personal simulation setup will also be given and downloadable on the website. Remind that it would be very grateful that a credit of the AMIDAS package and website could be given for using the output results.

4 Analyzing (real/pseudo-) data

The probably most important and useful design of the AMIDAS package and website is the ability of analyzing user-uploaded (real/pseudo-) data set(s) recorded in direct DM detection experiments *without* modifying the source code.

In this section, we describe the preparation of data files and the uploading/analyzing procedure on the AMIDAS website. A sample file for the uploaded data sets can be downloaded from the AMIDAS website.

4.1 Preparing data set(s)

As mentioned above, on the AMIDAS website users can find and download a sample file for the uploaded data sets. Note that, for comments a “0” (zero) *has to be used* at the beginning, and all words in the comment lines must be connected by, e.g. “_” (underscores). For instance,

Example: an uploaded data file

```
0 m_[chi]_=_100_GeV
0 sigma_[chi,_p]^SI=_1e-9_pb
0 .....
  :
  :

1 dataset, 58 events, 1628.565 kg-day:
  1      1      0.068 keV
  1      2      8.325 keV
    :
    :

2 dataset, 48 events, 1628.565 kg-day:
  2      1     11.743 keV
  2      2      1.824 keV
    :
    :
```

Note that, as shown in the above example, it is *unnecessary* to order the generated/recorded recoil energies ascendingly or descendingly in your uploaded data file(s). The AMIDAS package will order the events in each data set after reading these events. However, the generated/arecorded events with different target nuclei must be saved in *separate* files.

4.2 Uploading data file(s)

Once users have chosen the data type as “real (pseudo-) data” (Sec. 2.4.1), in the table for the experimental setup there will be one column for uploading users’ data file(s) (cf. Figure shown in Sec. 3.6.1).

Experimental setup

Target	Q_{\min} (keV)	Q_{\max} (keV)	b_1 (keV)	Uploading data file (max. 2 MB)
^{28}Si	<input type="text" value="0"/>	<input type="text" value="100"/>	<input type="text" value="10"/>	<input type="button" value="瀏覽..."/> 未選擇檔案。
^{76}Ge	<input type="text" value="0"/>	<input type="text" value="100"/>	<input type="text" value="10"/>	<input type="button" value="瀏覽..."/> 未選擇檔案。
<input type="text" value="50"/> data sets (max. 2,000)				

* Move the cursor onto a symbol for checking its definition.

Download: [sample for uploaded data sets](#)

Users can upload their data file(s) as usual. Note only that the maximal size of *each* uploaded file is *2 MB*. In addition, the numbers of data sets in *all* uploaded files must be *equal* or set as the *smallest* one⁸.

⁸Events in the “extra” data sets will however neglected in the analyses.

After that one or more data files have been uploaded and saved successfully, one extra column will appear in the experimental setup table to show the *original* name(s) of the uploaded data file(s).

Experimental setup

Target	Q_{\min} (keV)	Q_{\max} (keV)	b_1 (keV)	Uploading data file (max. 2 MB)	Uploaded data file
^{28}Si	<input type="text" value="0"/>	<input type="text" value="100"/>	<input type="text" value="10"/>	<input type="button" value="瀏覽..."/> 未選擇檔案。	
^{76}Ge	<input type="text" value="0"/>	<input type="text" value="100"/>	<input type="text" value="10"/>	<input type="button" value="瀏覽..."/> 未選擇檔案。	50events-50expts-Ge76-1.txt
<input type="text" value="50"/> data sets (max. 2,000)					

* Move the cursor onto a symbol for checking its definition.

Download: [sample for uploaded data sets](#)

Users can check whether the correct files have been uploaded for the corresponding targets and, if necessary, upload the correct file once again.

4.3 Analyzing uploaded data

As for simulations, after giving all the required information for the aimed analysis, users have one more chance to check their choices and the *original* name(s) of their data file(s), modify some of them and replace the uploaded data file(s), and then resubmit the whole setup, before they click the “Data analysis start” button to start the AMIDAS program.

Data analysis starting

Check the above setup

Remind that, in case that any required datum or *data file* is missed, this omission will be detected automatically after the (re)submission and users will be reminded of that with a *red* block around the options/table. Note that, while *all data* in a table with missed information will be *reset* to the default values and should therefore be checked and modified once again, users *only* need to upload the *missed* data file(s) and/or the *replacement(s)*.

Once all the required data and uploaded file(s) have been checked, users have only to click the “Data analysis start” button and wait for the analyzed results for a few minutes.

4.4 Output results (tables)

Reconstructed results will be presented in form(s) of table(s) and occasionally plot(s). For instance, for the reconstruction of the WIMP mass m_χ , one can obtain the following result⁹:

⁹Here two data files with the targets ^{28}Si and ^{76}Ge have been used. In each file there are 10 recorded data sets; in each data set there are 50 recorded events on average. The original setup for generating these pseudo-events can be found in Ref. [40].

Results

Reconstructed WIMP mass

$m_\chi \text{ (GeV}/c^2\text{)}$	$m_{\chi, \text{combined}}$
$m_{\chi, \text{rec}}$	122.76
$m_{\chi, \text{lo}}$	83.334
$m_{\chi, \text{hi}}$	196.37

The reconstructed WIMP mass and the upper and lower bounds of its 1σ statistical uncertainty. $m_{\chi, \text{combined}}$ has been estimated by the χ^2 -fitting defined in Eq. (51) of Ref. [4], which combines the estimators for $m_{\chi, n}$ and $m_{\chi, \sigma}$ estimated by Eqs. (34) and (40) of Ref. [4] with each other. The reconstructed WIMP mass $m_{\chi, \text{combined}}$ shown here have been corrected by the iterative Q_{max} -matching procedure described in Ref. [4].

Data

TXT: [amidas-140210-111108.txt](#)

In order to let users understand the output results more clearly and use them more conveniently, each output plot or table will be accompanied with a short description.

On the other hand, the original TXT file(s) of the reconstructed results with users' personal experimental setup will also be given and downloadable on the website. Remind that it would be very grateful that a credit of the **AMIDAS** package and website could be given for using the output results.

5 Bayesian analyses

In Ref. [5], we applied Bayesian analysis technique to the reconstruction of the one-dimensional velocity distribution function of Galactic WIMPs. This newest development is released for both of simulation and (real/pseudo-) data analysis on the **AMIDAS** website.

5.1 Fitting velocity distribution function of halo WIMPs

As the most crucial part in Bayesian analyses, users need to choose one fitting one-dimensional WIMP velocity distribution function. In the **AMIDAS-II** package, we offer so far five analytic forms for the fitting velocity distribution. Meanwhile, users also have the option to define (by uploading/typing) an analytic form of their favorite fitting one-dimensional velocity distribution:

Simulation setup II

Choose the fitting velocity distribution function

☐ Simple Maxwellian [analytic form](#)
☐ Modified Maxwellian [analytic form](#)
☒ One-parameter shifted Maxwellian [analytic form](#)
☐ Shifted Maxwellian [analytic form](#)
☐ Variated shifted Maxwellian [analytic form](#)
☐ User-defined fitting function with parameter(s)

Upload files: 未选择檔案

未选择檔案 (for drawing output plot)

Type definitions:

```

double Int_flv_Bayesian_fit_user(double vv, double aa, double bb, double cc)
{
    return -1.0;
}

double flv_Bayesian_fit_user(double vv, double aa, double bb, double cc, double x)
{
    flv_Bayesian_fit_user(x)
    = (x * x) * \
      - exp(-(x * x) / (cc * aa * aa)) \
      - exp(-(bb * bb) / (cc * 1.0 * 1.0)) ** cc
  
```

(for drawing output plot)

Download: [sample for a user-defined fitting function, for drawing output plots](#)

Note here that, firstly, for checking the analytic forms of the intrinsically defined fitting velocity distributions, users can hover the cursor onto “analytic form”; users can also click the “analytic form” to open a new webpage with more detailed information and useful references. Secondly, as reminded on the website, the second uploaded file/typing area is *only* for drawing output plot(s) of the Bayesian reconstructed one-dimensional WIMP velocity distribution function. Additionally, for defining users own fitting velocity distribution, the number of the fitting parameters must be set simultaneously on the website¹⁰.

In this subsection, we give first the definitions of the five default fitting one-dimensional WIMP velocity distribution functions in the **AMIDAS-II** package. Then we will describe how to (modify these definitions to) define user’s own fitting velocity distribution.

5.1.1 Default fitting one-dimensional WIMP velocity distribution functions

Except of three default one-dimensional WIMP velocity distributions for generating WIMP signal events (see Sec. 3.3.1), in Ref. [5] two useful variations of the shifted Maxwellian velocity distribution have also been considered. Hence, so far users have five options for the fitting one-dimensional WIMP velocity distribution defined intrinsically in the **AMIDAS-II** package [5]:

1. the simple Maxwellian velocity distribution function $f_{1,\text{Gau}}(v)$ [1];
2. the modified Maxwellian velocity distribution function $f_{1,\text{Gau},k}(v)$ [27, 28, 29, 30];
3. the one-parameter shifted Maxwellian velocity distribution function

$$f_{1,\text{sh},v_0}(v) = \frac{1}{\sqrt{\pi}} \left(\frac{v}{v_0 v_e} \right) \left[e^{-(v-v_e)^2/v_0^2} - e^{-(v+v_e)^2/v_0^2} \right], \quad (26)$$

with

$$v_e = 1.05 v_0 \quad (27)$$

is the time-averaged (time-independent) Earth’s velocity in the Galactic frame [5];

¹⁰So far the **AMIDAS-II** package allows to fit/scan maximal *three* fitting parameters.

4. the shifted Maxwellian velocity distribution function $f_{1,\text{sh}}(v)$ [1, 31];
5. the variated shifted Maxwellian velocity distribution function [5]

$$f_{1,\text{sh},\Delta v}(v) = \frac{1}{\sqrt{\pi}} \left[\frac{v}{v_0(v_0 + \Delta v)} \right] \left\{ e^{-[v-(v_0+\Delta v)]^2/v_0^2} - e^{-[v+(v_0+\Delta v)]^2/v_0^2} \right\}, \quad (28)$$

where

$$\Delta v \equiv v_e - v_0 \quad (29)$$

is the difference between v_0 and the time-dependent Earth’s velocity in the Galactic frame $v_e(t)$.

5.1.2 User-defining the fitting one-dimensional WIMP velocity distribution function

For defining users’ favorite fitting velocity distribution function to include into the **AMIDAS-II** package, one has to distinguish two cases: an analytic form for the *integral* over the fitting velocity distribution *exists* or *not*.

As an example, we give here the **AMIDAS-II** code for the (integral over the) simple Maxwellian velocity distribution function for the Bayesian fitting process. The codes for the (integral over the) other intrinsically defined “fitting” one-dimensional velocity distribution functions will be given in Appendix C.1 for users’ references.

For the case that an analytic form for the integral over the fitting velocity distribution exists, users have to define this integral with the name of “Int_f1v_Bayesian_fit_user” *first*:

Code 10: “Defined” integral over the fitting simple Maxwellian velocity distribution
 $\int f_{1,\text{Gau}}(v) dv$

```
double Int_f1v_Bayesian_fit_user(double vv, double aa, double bb, double cc)
{
    return
        erf(vv / aa)
        - (2.0 / sqrt(M_PI)) *
        (vv / aa) *
        exp(-(vv * vv) / (aa * aa));
}
```

And then define the fitting velocity distribution function with the name of “f1v_Bayesian_fit_user” *itself*:

Code 11: Fitting simple Maxwellian velocity distribution $f_{1,\text{Gau}}(v)$

```
double f1v_Bayesian_fit_user(double vv, double aa, double bb, double cc, double N_f)
{
    return
        ( (4.0 / sqrt(M_PI)) *
          ( (vv * vv) / (aa * aa * aa) ) / v_U *
          exp(-(vv * vv) / (aa * aa)) ) /
        ( Int_f1v_Bayesian_fit_user(v_max, aa, bb, cc)
          - Int_f1v_Bayesian_fit_user(0.0, aa, bb, cc) );
}
```

Note that, firstly, the function `Int_f1v_Bayesian_fit_user` for the integral has *always four* parameters: the velocity `vv` and *three* fitting parameters `aa`, `bb` and `cc`. It doesn't matter how many fitting parameters (only one or two or all three) are actually used. Meanwhile, the function `f1v_Bayesian_fit_user` for the velocity distribution itself has *always five* parameters: except of `vv`, `aa`, `bb` and `cc`, an extra parameter `N_f` has to be included, which plays the role of the normalization constant needed *only* for the case that an analytic form for the integral over the fitting velocity distribution does *not* exist (see below).

On the other hand, once it is too complicated or even impossible to give an analytic form for the integral over the fitting velocity distribution, one can *first* define it as “-1.0”:

Code 12: “Undefined” integral over the fitting simple Maxwellian velocity distribution
 $\int f_{1,\text{Gau}}(v) dv$

```
double Int_f1v_Bayesian_fit_user(double vv, double aa, double bb, double cc)
{
    return -1.0;
}
```

And then define the *main (velocity-dependent)* part of the fitting velocity distribution function:

Code 13: Fitting simple Maxwellian velocity distribution $f_{1,\text{Gau}}(v)$

```
double f1v_Bayesian_fit_user(double vv, double aa, double bb, double cc, double N_f)
{
    return
        ( ( (vv * vv) / (aa * aa * aa) ) *
          exp(-(vv * vv) / (aa * aa) ) ) /
        N_f;
}
```

Then the **AMIDAS-II** package will estimate the *fitting-parameter dependent* normalization constant N_f numerically *point-by-point* during the Bayesian fitting (scanning) procedure.

Additionally, users have to set the symbol and unit of *each* fitting parameter, which will be used in the output result files and plots, with the names of “ `a_Bayesian_Symbol`” and “ `a_Bayesian_Unit`” in the *same* uploaded file:

Code 14: Symbol and unit of the (first) fitting parameter

```
a_Bayesian_Symbol = "v_0";
a_Bayesian_Unit   = "km/s";
```

Finally, as for drawing the generating velocity distribution (Sec. 3.3), an *extra* file is required for defining the “fitting” one-dimensional velocity distribution function *itself* (the integral over it is not needed anymore) for drawing the Bayesian reconstructed/fitted velocity distribution in the output plots. Remind that, since the **Gnuplot** package has been adopted in **AMIDAS** for drawing output plots, the velocity distribution function defined in this file must be written in the syntax of **Gnuplot** with the name of “`f1v_Bayesian_fit_user(x)`”. Below we give our definition for the simple Maxwellian velocity distribution with and without the known analytic form of the integral over it as examples. The definitions for the other intrinsically defined fitting one-dimensional WIMP velocity distribution function are given in Appendix C.1 for users' references.

Once the analytic form of the integral over the velocity distribution function is known, we have


```
M_PI = 3.141593

f1v_Bayesian_fit_user(x) \
= (4.0 / sqrt(M_PI)) * \
  ( (x * x) / (aa * aa * aa) ) * \
  exp(-(x * x) / (aa * aa) )
```

Code 16: Fitting simple Maxwellian velocity distribution $f_{1,\text{Gau}}(v)$ (for Gnuplot)

$$= \frac{f1v_Bayesian_fit_user(x)}{(x * x) / (aa * aa * aa)} * \exp(-(x * x) / (aa * aa))$$

5.2 Distribution function for describing the statistical uncertainty

In this subsection, we discuss the second key factor required in Bayesian analyses: the distribution function for describing the statistical uncertainty on the analyzed/fitted data points, which will in turn be used in the likelihood function (for more details, see Ref. [5] and references therein). In the **AMIDAS-II** package, we offer so far two analytic forms for the statistical-uncertainty distribution function. Meanwhile, users also have the option to define (by uploading/typing) an analytic form of their needed uncertainty distribution:

Choose the distribution function for describing the statistical uncertainty

☐ Poisson-distributed

☒ Gaussian-distributed

☐ User-defined distribution

Upload file:

Type definition:

```
double Bayesian_DF_user(double y_th, double y_rec, double y_lo, double y_hi)
{
    if (y_rec >= y_th)
    {
        return
            exp(-y_rec) * pow(y_rec, y_th) / (y_lo - y_hi) * (1 - exp(-y_lo)) * (1 - exp(-y_hi))
    }
}
```

Download: [sample for a user-defined distribution function](#)

Note however here that, *only* the option for a “Gaussian–distributed” statistical uncertainty is considered for our Bayesian reconstruction of the one–dimensional WIMP velocity distribution function. Thus the other one option for a “Poisson–distributed” statistical uncertainty as well as the uploading/typing areas are “unavailable” (locked) currently.

¹¹We set $N_f = 1$ at first and evaluate numerically the (reciprocal of the) integral over `f1v_Bayesian_fit_user(vv, aa, bb, cc, N_f = 1.0)` by using the *Simpson's rule*.

Nevertheless, as the users' guide to the **AMIDAS(-II)** package and website, in this subsection, we give still first the definition of the intrinsically defined Gaussian uncertainty distribution function used for the Bayesian reconstruction of $f_1(v)$ [5]. Then we will describe how to (modify this definition to) define user's own statistical-uncertainty distribution.

5.2.1 Default statistical-uncertainty distribution functions

As the most commonly considered cases, in the **AMIDAS-II** package we define two statistical-uncertainty distribution functions:

1. Poisson-distributed

$$\text{Poi}(x_i, y_i; a_j, j = 1, 2, \dots, N_{\text{Bayesian}}) = \frac{f^{y_i}(x_i; a_j) e^{-f(x_i; a_j)}}{y_i!}, \quad (30)$$

where (x_i, y_i) for $i = 1, 2, \dots, N$ denote the N analyzed/fitted data points and $f(x; a_j)$ is the theoretically predicted/fitting function with the fitting parameters a_j , for $j = 1, 2, \dots, N_{\text{Bayesian}}$;

2. Gaussian-distributed

$$\text{Gau}(x_i, y_i, \sigma(y_i); a_j, j = 1, 2, \dots, N_{\text{Bayesian}}) = \frac{1}{\sqrt{2\pi} \sigma(y_i)} e^{-[y_i - f(x_i; a_j)]^2 / 2\sigma^2(y_i)}, \quad (31)$$

where $(x_i, y_i \pm \sigma(y_i))$ for $i = 1, 2, \dots, N$ denote the N analyzed/fitted data points with the (measurement) uncertainties on $y_i, \sigma(y_i)$.

5.2.2 User-defining the statistical-uncertainty distribution function

As an example, we give here the **AMIDAS-II** code for the (double-)Gaussian statistical-uncertainty distribution function used for our Bayesian reconstruction procedure [5] (with the name of "Bayesian_DF_user"). The code for the other intrinsically defined (i.e. Poisson, so far) statistical-uncertainty distribution function will be given in Appendix C.2 for users' references.

Code 17: (Double-)Gaussian statistical-uncertainty distribution $\text{Gau}(x_i, y_i, y_{i,(\text{lo}, \text{hi})}; y_{i, \text{th}})$

```
double Bayesian_DF_user(double y_th, double y_rec, double y_lo, double y_hi)
{
    if (y_rec >= y_th)
    {
        return
            exp(-(y_rec - y_th) * (y_rec - y_th) /
                (2.0 * (y_rec - y_lo) * (y_rec - y_lo))) /
            (sqrt(2.0 * M_PI) * (y_rec - y_lo));
    }

    else
    if (y_rec < y_th)
    {
        return
            exp(-(y_rec - y_th) * (y_rec - y_th) /
                (2.0 * (y_hi - y_rec) * (y_hi - y_rec))) /
            (sqrt(2.0 * M_PI) * (y_hi - y_rec));
    }
}
```

Note here that we consider in the **AMIDAS-II** package an *asymmetric* distribution of the statistical uncertainty. For the case that the analyzed/fitted data point is larger (smaller) than the theoretical value estimated from the fitting (WIMP velocity distribution) function, we take the 1σ lower (upper) (statistical) uncertainty as the uncertainty $\sigma(y_i)$ in Eq. (31). Remind that, a sample file for the **AMIDAS** code can be downloaded from the **AMIDAS** website.

5.3 Distribution function of each fitting parameter

In Bayesian analyses, one needs to give a (probability) distribution function for describing no/a prior knowledge about each fitting parameter. In the **AMIDAS-II** package, we offer so far three probability distribution functions for each fitting parameter for describing the prior knowledge about it. Meanwhile, users also have the option to define (by uploading/typing) an analytic form of their favorite probability distribution:

Choose the distribution function about the first fitting parameter v_0

☒ Flat-distributed
☐ Poisson-distributed
☐ Gaussian-distributed
☐ User-defined distribution

Upload file: 浏览... 未選擇檔案

Type definition:

```
double Bayesian_DF_a_user(double aa)
{
    return 1.0;
}
```

Download: sample for a user-defined [flat](#)/[Poisson](#)/[Gaussian](#) distribution function

Note that, since the statistical uncertainties on the default fitting parameters used for our Bayesian reconstruction of $f_1(v)$ are in principle *Gaussian*, the option for a “Poisson–distributed” probability distribution function is “unavailable” (locked) here. It is however possible to use the intrinsically defined “Poisson” probability distribution (unlocked), once users choose to define their own fitting WIMP velocity distribution function (Sec. 5.2).

In this subsection, we give first the definitions of the three intrinsically defined probability distribution functions for each fitting parameter in the **AMIDAS-II** package. Then we will describe how to (modify these definitions to) define user’s favorite probability distribution.

5.3.1 Default distribution functions for the fitting parameters

As the most commonly considered cases, in the **AMIDAS-II** package we define three probability distribution functions:

1. Flat–distributed:

$$p_{i,\text{flat}}(a_i) = 1, \quad \text{for } a_{i,\text{min}} \leq a_i \leq a_{i,\text{max}}, \quad (32)$$

where $a_{i,\text{min/max}}$ denote the minimal and maximal bounds of the scanning interval of the fitting parameter a_i ;

2. Poisson–distributed:

$$p_{i,\text{Poi}}(a_i; \mu_{a,i}) = \frac{\mu_{a,i}^{a_i} e^{-\mu_{a,i}}}{a_i!}, \quad (33)$$

with the expected value $\mu_{a,i}$ of the fitting parameter a_i ;

3. Gaussian-distributed:

$$p_{i,\text{Gau}}(a_i; \mu_{a,i}, \sigma_{a,i}) = \frac{1}{\sqrt{2\pi} \sigma_{a,i}} e^{-(a_i - \mu_{a,i})^2 / 2\sigma_{a,i}^2}, \quad (34)$$

with the expected value $\mu_{a,i}$ of and the 1σ uncertainty $\sigma_{a,i}$ on the fitting parameter a_i .

5.3.2 User-defining the distribution function for the fitting parameters

As an example, we give here the **AMIDAS-II** code for the Gaussian probability distribution function used in our Bayesian analysis procedure [5] (with the name of “**Bayesian_DF_a_user**”). The codes for the other intrinsically defined probability distribution functions will be given in Appendix C.3 for users’ references.

Code 18: Gaussian probability distribution for the fitting parameter $p_{i,\text{Gau}}(a_i; \mu_{a,i}, \sigma_{a,i})$

```
double a_ave_Bayesian_user      = 2.0;
double sigma_a_ave_Bayesian_user = 0.5;

double Bayesian_DF_a_user(double aa)
{
    return
        exp(-(aa - a_ave_Bayesian_user) *
            (aa - a_ave_Bayesian_user) /
            (2.0 *
              sigma_a_ave_Bayesian_user *
              sigma_a_ave_Bayesian_user ) ) /
        (sigma_a_ave_Bayesian_user * sqrt(2.0 * M_PI));
}
```

Note that there is *only one* function parameter in the (user-defined) probability distribution function and the parameter name must be “aa”, “bb” or “cc” corresponding to the first, second or the third fitting parameter. Remind also that, a sample file for the **AMIDAS** code can be downloaded from the **AMIDAS** website.

5.3.3 Bounds of the scanning interval of each fitting parameter

For doing Bayesian fitting (scanning), users need to set the lower and upper bounds of the scanning range for each fitting parameter:

Give bounds of the scanning interval of the first fitting parameter v_0

Lower bound:	160	km/s
Upper bound:	300	km/s

Moreover, once the probability distribution function of one fitting parameter has been chosen as, e.g. Gaussian-distributed, it will be required automatically to set also the expected value and the standard deviation (uncertainty) of this parameter:

Give bounds of the scanning interval of the first fitting parameter v_0

Lower bound: 160 km/s
Upper bound: 300 km/s
Expected value: 230 km/s
Standard deviation: 20 km/s

On the other hand, for the case that user-defined probability distribution function has been used for one fitting parameter, users have also to give the notation and the unit of this parameter for the output files and plots:

Give bounds of the scanning interval of the first fitting parameter

Notation: v_0
Unit: km/s (Use "NU" for a "dimensionless" parameter)
Lower bound: 160
Upper bound: 300
Expected value: 230
Standard deviation: 20

Note that, as remind on the website, for a “dimensionless” parameter (e.g. the ratio v_e/v_0 and the power index k) it is required to use “NU” or “nu” in the input cell.

5.4 Scanning the parameter space

Finally, as the last information for our Bayesian analysis procedure, users have the opportunity to choose different scanning method.

5.4.1 Method for scanning the parameter space

(So far) in the AMIDAS-II package we programmed *three* different scanning methods:

Choose the method for scanning the parameter space

- ☒ Scan the whole parameter space regularly
- ☐ Scan the whole parameter space roughly and then the neighborhood of the valid points more precisely
- ☐ Scan the whole parameter space roughly and then the neighborhood of the (almost) valid points more precisely

Here the second option “scan the whole parameter space roughly and then the neighborhood of the valid points more precisely” means that AMIDAS-II scans the whole parameter space to find the valid points; after that one valid point is found, AMIDAS-II will scan *immediately* and *randomly* the neighborhood of this point for finding a better (more possible) point¹².

For a finer scanning, the third option above “scan the whole parameter space roughly and then the neighborhood of the (almost) valid points more precisely” let AMIDAS-II scan *immediately*

¹²Once a valid point, called the “starting” point with the value a_i^* , is found, we pick *randomly* one point in the range of $[\max(a_i^* - \Delta_{a_i}, a_{i,\min}), \min(a_i^* + \Delta_{a_i}, a_{i,\max})]$, where $\Delta_{a_i} = (a_{i,\max} - a_{i,\min}) / (\text{the number of scanning points})$. We take this point if it is better and set this point as the next starting point for running this fine scanning process further; if this point is however *invalid*, we through it away, use the old starting point to pick another checking point and run this fine scanning process. Note that, in the fine scanning process, we set *twice more* scanning points on one parameter-axis around the first starting point (see Sec. 5.4.2 for more details).

and *randomly* the neighborhood of the point for finding a better point, once this point is valid or almost valid¹³.

5.4.2 Number of scanning points for one fitting parameter

Users can set the number of scanning points for one fitting parameter:

Give the number of scanning points for one fitting parameter

Number of scanning points:

Note that this point number will be used for the regular scanning (option 1 in Sec. 5.4.1) as well as for the “first step” of the rough scanning (options 2 and 3). For the finer scanning around the (almost) valid points taken by the first step, *doubled* points (picked randomly around each (almost) valid point) for one parameter will be checked¹⁴. Hence, the default number of scanning points (for one fitting parameter) needs to set different according to the choice of the scanning method.

6 Summary

In this paper, we give a detailed user’s guide to the **AMIDAS** (A Model–Independent Data Analysis System) (package and) website, which is developed for online simulations and data analyses for direct Dark Matter detection experiments and phenomenology.

AMIDAS(–II) has the ability to do full Monte Carlo simulations as well as to analyze (real/pseudo–) data sets either generated by another event generating programs or recorded in direct DM detection experiments. Recently, the whole (**AMIDAS** package and website) system has been upgraded to the second phase: **AMIDAS–II**, for including the new developed Bayesian analysis technique.

Users can run all functions and adopt the (default) input setup used in our (earlier) works [4, 5, 6, 7, 8, 23, 24, 25, 26, 40, 39] for their simulations as well as analyzing their own (real/pseudo–) data sets. The use of the **AMIDAS** website for users’ simulations and data analyses has been explained step–by–step with plots in this paper. The preparations of function/data files to upload for simulations and data analyses have also been described.

Moreover, for more flexible and user–oriented use, users have the option to set their own target nuclei as well as their favorite/needed (fitting) (one–dimensional) WIMP velocity distribution function, (more suitable) elastic nuclear form factors for the SI and SD WIMP–nucleus cross sections and different probability distribution functions needed in the Bayesian reconstruction procedure. As examples, the **AMIDAS(–II)** codes for all user–uploadable functions are given in Secs. 3 and 5 as well as Appendix B and C.

¹³For the Bayesian reconstruction of the one–dimensional WIMP velocity distribution function, we set in **AMIDAS–II** that the value of the posterior probability distribution function at the *checking* point is *larger than* 90% of “so far” largest one, since we only need to find the point, in which the value of the posterior probability distribution function is the largest.

¹⁴This means that, once we set e.g. 100 scanning points (for one fitting parameter), 200 more points (in one parameter–axis) will be scanned in the neighborhood around each (almost) valid point. This means in turn that, for a fitting (velocity distribution) function with “three” parameters to scan, for “each” (almost) valid point, **AMIDAS–II** needs to check $\sim 8,000,000$ more points (and perhaps do 1 or $2 \times \sim 8,000,000$ more times three–dimensional numerical integrations for the normalization constant of the fitting velocity distribution!).

In summary, up to now all basic functions of the **AMIDAS** package and website have been well established. Hopefully this new tool can help our theoretical as well as experimental colleagues to understand properties of halo WIMPs, offer useful information to indirect DM detection as well as collider experiments, and finally discover (the mystery of) Galactic DM particles.

Acknowledgments

The author would like to thank James Yi-Yu Liu for useful discussion about the basic idea of including user-defined functions into the source code. The author appreciates the ILIAS Project and the Physikalisches Institut der Universität Tübingen for kindly providing the opportunity of the collaboration and the technical support of the **AMIDAS** website. The author would also like to thank the friendly hospitality of the Institute of Physics, National Chiao Tung University, the Graduate School of Science and Engineering for Research, University of Toyama, the Institute of Modern Physics, Chinese Academy of Sciences and the Center for High Energy Physics, Peking University, where part of this work was completed. This work was partially supported by the National Science Council of R.O.C. under contracts no. NSC-98-2811-M-006-044 and no. NSC-99-2811-M-006-031.

A Intrinsically defined constants

In this section, we list all constants as well as the reduced mass $m_{r,N}$ in Eq. (7) and the transformation constant α in Eq. (6) defined intrinsically in the **AMIDAS** package for users' reference. Users could use these intrinsically defined constants and functions directly and/or include them into their own defined functions.

A.1 Constants and translators

The physical constants and needed translators defined in the **AMIDAS** package are in *natural units* as following:

Code A1: Physical constants and needed translators

```
double c    = 1.0;

double v_U = c / (2.9979246 * 1e5);

double m_U = 1e6 / (c * c);
double m_p = 0.938272 * m_U;

double G_F = 1.166379 * ( 1e-5 / (1e6 * 1e6) ) * (c * c * c);

double fm_U = c / (0.197327 * 1e6);
double pb_U = 1e-36 * (1e13 * fm_U) * (1e13 * fm_U);

double s_U = 1.0 / (6.582119 * 1e-19);
double kg_day_U = (m_U / (1.78266 * 1e-27)) * (86400.0 * s_U);

double rho_U = m_U / ( (1e13 * fm_U) * (1e13 * fm_U) * (1e13 * fm_U) );

double omega = 2.0 * M_PI / 365.0;
```

The meanings of these constants and translators are

Symbol	Meaning	Remarks
v_U	The velocity translator from km/s to $c = 2.997\,924\,58 \times 10^5$ km/s	[41]
m_U	The mass translator from GeV/c^2 to keV/c^2	
m_p	The proton mass $m_p = 938.272\,046(21)$ MeV/ c^2	[41]
G_F	The Fermi coupling constant $G_F = 1.166\,378\,7(6) \times 10^{-5}$ $\text{GeV}^{-2}(\hbar c)^3$	[41]
fm_U	The length translator from 1 fm = 10^{-15} m to c ($\hbar c = 0.197\,326\,9718(44)$ GeV fm)	[41]
pb_U	The area translator from 1 pb = 10^{-36} cm ² to the natural units	
s_U	The time translator from 1 s to the natural units ($\hbar = 6.582\,119\,28(15) \times 10^{-25}$ GeV s)	[41]
kg_day_U	The exposure translator from 1 kg-day to the natural units (1 GeV/ c^2 = $1.782\,661\,845(39) \times 10^{-27}$ kg)	[41]
rho_U	The density translator from 1 GeV/ c^2 /cm ³ to the natural units	
omega	$\omega = 2\pi/365$ needed in Eq. (4)	

A.2 The reduced mass $m_{r,N}$ and the transformation constant α

All masses in the AMIDAS code are in the unit of GeV/c^2 . First, the mass of the target nucleus is defined as a function of the atomic mass number A by

Code A2: Nuclear mass $m_N(A)$

```
double m_N(int A)
{
    return m_p * A * 0.99;
}
```

Here the mass difference between a proton and a neutron has been neglected. And the reduced mass between the WIMP mass and “something” is given by

Code A3: Reduced mass between the WIMP mass and “something”

```
double mchi_r(double mchi, double mx)
{
    return mchi * mx / (mchi + mx);
}
```

Hence, the reduced mass of the WIMP mass with the proton (nucleon) mass and with the mass of the target nucleus, $m_{r,N}$ in Eq. (7), are defined as

Code A4: Reduced mass $m_{r,p}(m_\chi)$

```
double mchi_rp(double mchi)
{
    return mchi_r(mchi * m_U, m_p);
}
```


and

Code A5: Reduced mass $m_{r,N}(m_\chi, A)$

```
double mchi_rN(double mchi, int A)
{
    return mchi_r(mchi * m_U, m_N(A));
}
```

Finally, the transformation constant α defined in Eq. (6) can also be given as a function of m_χ and A as:

Code A6: transformation constant $\alpha(m_\chi, A)$

```
double alpha(double mchi, int A)
{
    return sqrt(m_N(A) / 2.0) / mchi_rN(mchi, A);
}
```

A.3 Declared variables and constants

Here we list the *short-named* variables and constants declared in the AMIDAS(-II) package. Note that these names should be *avoided* to use in the user-defined function, except that you know their exact meanings and can apply them properly; otherwise, the AMIDAS(-II) package might not work correctly.

Declared short-named variables and constants

```
a_point, a_pointNo, a_subpoint, a_subpointNo, an_in, ap_in, Ax, AX, AX_max, AX_str,
b_point, b_pointNo, b_subpoint, b_subpointNo, bar_Q, bar_Q_win, bar_QQ, bar_QQ_win,
bgevent_ratio, bgevent_ratio_common, c_point, c_pointNo, c_subpoint, c_subpointNo,
calN, chi2_mchi_min, CnX, CnX_n, CnX_n_sh, CnX_sh, cov_InX, cp, cp_sh, CpX, CpX_p,
CpX_p_sh, CpX_sh, Delta_Qt, Delta_Qt_tmp, Delta_Qtp, Delta_Qtp_tmp, Delta_t,
Delta_t_tmp, delta_Xbinwidth, delta2_k, delta2_k_win, delta2_Q, delta2_Q_win,
Epsilon, error_mchi_in_fixed, error_sigmaSIp_in, event, eventNo_bg, eventNo_bg_ave,
eventNo_max, eventNo_Q, eventNo_Q_win, eventNo_sg, eventNo_sg_ave, eventNo_tot,
eventNo_tot_ave, exptNo_common, exptNo_max, fp2_algo, fp2_input, fp2_th, fp2_tmp,
FQmin_SD_TX, FQmin_SI_TX, FQmin_TX, FQ_TS_const, hn, hn_median, hn_tmp,
Intf1v_Gau_k, Intf_dim, Intf_function_Simpson, Intf_pointNo, InX, JJ, k, k_flv,
k_win, ka, kappa, kappa_hi, kappa_lo, kappa_win, kn, kn_median, kn_tmp, lambda, ln,
ln_median, ln_tmp, m_N_TX, mchi_assumed, mchi_in, mchi_in_fixed, mchi_solve,
mchi_solve_hi, mchi_solve_lo, mchi_tmp, mchiNo, mchiNo_max, mm, n_expt, n_mchi,
n_plot, n_point, n_ranap, NmX, nn, nn_max, nn_win, nTX, nTXh, nTXl, nTXx, nTXxn,
nTXxp, nTXy, nTXyn, nTXyp, nTXz, order_sigmaSIp_in, pm, pointNo, Qbin, Qbin_1_common,
Qbin_hi, Qbin_lo, Qbin_max, QbinNo, QbinNo_common, QbinNo_max, Qbinwidth, Qmax,
Qmax_algo, Qmax_bg_str, Qmax_bgwindow, Qmax_bgwindow_common, Qmax_gen, Qmax_Int,
Qmax_InX_Int, Qmax_kin, Qmax_set, Qmax_set_common, Qmax_str, Qmax_win, Qmid, Qmid_sh,
Qmid_win, Qmid_win_sh, Qmin, Qmin_bg_str, Qmin_bgwindow, Qmin_bgwindow_common,
Qmin_gen, Qmin_Int, Qmin_set, Qmin_set_common, Qmin_str, Qmin_win, QminXInX,
qqR_1_min, qqR_1_max, QQ_SD_max_TX, QQ_SD_min_TX, Qtp, Qvin, Qwin, Qwin_max, QwinNo,
QwinNo_max, Qwinwidth, R_Simpson, R_TX, r_win, ranap_in, ranap_in_fixed, ranap_SD,
ranap_SD_sh, ranap_tmp, ranapNo, ranapNo_max, ratio_eventNo_bg_ave,
ratio_eventNo_sg_ave, ratio_eventNo_tot_ave, ratio_Xbinwidth, ratio_Ybinwidth,
ratio_Zbinwidth, rfnfp_in_fixed, rho, rho_0, rho0_input, rho0_tmp, RJ, RJ_sh, RJX,
rlh, rmin_bgwindow, rmin_gen, rminXInX, Rn, Rn_sh, RnX, Rsigma, Rsigma_sh,
```

```

rsigmaSDnSI_th, rsigmaSDnSI_tmp, rsigmaSDpSI_th, rsigmaSDpSI_tmp, RsigmaX, run,
run_Intf, run_Intf_max, runi, runj, runk, scan_pointNo, scan_subpointNo, sh,
sigmaSIp_in, Sn, Snp, sol, Sp, Spn, subpointNo, sum_Q, sum_Q_win, sum_QQ, sum_QQ_win,
t_end, t_expt, t_p, t_start, tbin, tbinNo, tbinNo_common, tbinNo_max, tbinwidth, tmax,
tmax_Int, tmid, tmin, tmin_Int, TX, TXNo, TXNo_max, v_0, v_esc, v_max, Xbin, XbinNo,
XbinNo_max, Xbinwidth, Xbinwidth_1, Xbinwidth_common, Xmax, Xmax_binning, Xmax_Intf,
Xmax_plot, Xmid, Xmin, Xmin_binning, Xmin_Intf, Xmin_plot, Ybin, YbinNo, YbinNo_max,
Ybinwidth_1, Ymax_binning, Ymax_Intf, Ymax_plot, Ymin_binning, Ymin_Intf, Ymin_plot,
Zbin, ZbinNo, ZbinNo_max, Zbinwidth_1, Zmax_binning, Zmax_Intf, Zmin_binning,
Zmin_Intf, ZX

```

B Intrinsically defined functions

In this section, we give all codes for intrinsically defined velocity distribution functions given in Sec. 3.3, elastic nuclear form factors given in Sec. 3.4 as well as simple artificial background spectrum given in Sec. 3.5 in the `AMIDAS(-II)` package.

B.1 One-dimensional WIMP velocity distribution function

In this subsection, we give first the codes for intrinsically defined velocity distribution functions given in Sec. 3.3 in the `AMIDAS-II` package.

B.1.1 Simple Maxwellian velocity distribution $f_{1,\text{Gau}}(v)$

According to Eq. (1), we define

Code A7: Simple Maxwellian velocity distribution $f_{1,\text{Gau}}(v)$

```

double f1v_Gau(double vv)
{
    return
        (4.0 / sqrt(M_PI)) *
        ( (vv * vv) / (v_0 * v_0 * v_0) ) / v_U *
        exp(-(vv * vv) / (v_0 * v_0) );
}

```

Then, since

$$\int_{\alpha\sqrt{Q}}^{\infty} \left[\frac{f_{1,\text{Gau}}(v)}{v} \right] dv = \frac{2}{\sqrt{\pi}} \left(\frac{1}{v_0} \right) e^{-\alpha^2 Q/v_0^2}, \quad (\text{A1})$$

we can define

Code A8: Integral over the simple Maxwellian velocity distribution $\int_{\alpha\sqrt{Q}}^{\infty} [f_{1,\text{Gau}}(v)/v] dv$

```

double Intf1v_v_Gau(double mchi, int A, double QQ)
{
    return
        ( 2.0 / sqrt(M_PI) / (v_0 * v_U) ) *
        exp(-alpha(mchi, A) * alpha(mchi, A) * QQ /
            ( (v_0 * v_U) * (v_0 * v_U) ) );
}

```

On the other hand, for drawing output plots, we have

Code A9: Simple Maxwellian velocity distribution $f_{1,\text{Gau}}(v)$ (for Gnuplot)

```
M_PI = 3.141593

f1v_Gau(x) \
= (4.0 / sqrt(M_PI)) * \
  ( (x * x) / (v_0 * v_0 * v_0) ) * \
  exp(-(x * x) / (v_0 * v_0))
```

and

Code A10: Integral over the simple Maxwellian velocity distribution $\int_{\alpha\sqrt{Q}}^{\infty} [f_{1,\text{Gau}}(v)/v] dv$ (for Gnuplot)

```
c = 1.0
m_U = 1e6 / (c * c)
v_U = c / (2.9979246 * 1e5)

m_N = (0.938272 * m_U) * AX * 0.99
m_rN = (m_chi * m_U) * m_N / (m_chi * m_U + m_N)

alpha = sqrt(m_N / (2.0 * m_rN * m_rN))

Intf1v_v_Gau(x) \
= (2.0 / sqrt(M_PI) / (v_0 * v_U)) * \
  exp(-alpha * alpha * x / ( (v_0 * v_U) * (v_0 * v_U) ))
```

Remind that firstly, “\” (backslash) must be used in order to let the including of the definition(s) given in this file into the other intrinsic commands correctly. Secondly, three flexible-kept parameters: **AX**, **m_chi** and **v_0** will be read directly from the users’ initial (simulation/data analysis) setup set earlier on the website.

B.1.2 Modified Maxwellian velocity distribution $f_{1,\text{Gau},k}(v)$

According to Eq. (2), we can obtain the integral over $f_{1,\text{Gau},k}(v)$ and then define the normalization constant N_f in the **AMIDAS-II** package as follows.

For $k = 1$, we have

$$\begin{aligned} \int_0^{v_{\max}} f_{1,\text{Gau},k=1}(v) dv &= \frac{1}{N_{f,k=1}} \int_0^{v_{\max}} v^2 \left(e^{-v^2/v_0^2} - e^{-v_{\max}^2/v_0^2} \right) dv \\ &= \frac{1}{N_{f,k=1}} \left[\frac{\sqrt{\pi}}{4} v_0^3 \operatorname{erf} \left(\frac{v_{\max}}{v_0} \right) - v_{\max} \left(\frac{v_0^2}{2} + \frac{v_{\max}^2}{3} \right) e^{-v_{\max}^2/v_0^2} \right] \\ &= 1, \end{aligned} \tag{A2}$$

and then can define N_f by

Code A11: Normalization constant N_f of $f_{1,\text{Gau},k}(v)$ for $k = 1$

```

Int_f1v_Gau_k
= ( (sqrt(M_PI) / 4.0) *
    (v_0 * v_0 * v_0) *
    erf(v_max / v_0)
  - v_max *
    ( v_0 * v_0 / 2.0
    + v_max * v_max / 3.0 ) *
    exp(-(v_max * v_max) / (v_0 * v_0)) ) *
v_U;

```

For $k = 2$, we have

$$\begin{aligned}
& \int_0^{v_{\max}} f_{1,\text{Gau},k=2}(v) dv \\
&= \frac{1}{N_{f,k=2}} \int_0^{v_{\max}} v^2 \left(e^{-v^2/2v_0^2} - e^{-v_{\max}^2/2v_0^2} \right)^2 dv \\
&= \frac{1}{N_{f,k=2}} \left[\frac{\sqrt{\pi}}{4} v_0^3 \operatorname{erf}\left(\frac{v_{\max}}{v_0}\right) - v_{\max} \left(\frac{v_0^2}{2} - \frac{v_{\max}^2}{3} - 2v_0^2 \right) e^{-v_{\max}^2/v_0^2} \right. \\
&\quad \left. - \sqrt{2\pi} v_0^3 \operatorname{erf}\left(\frac{v_{\max}}{\sqrt{2}v_0}\right) e^{-v_{\max}^2/2v_0^2} \right] \\
&= 1,
\end{aligned} \tag{A3}$$

and then can define N_f by

Code A12: Normalization constant N_f of $f_{1,\text{Gau},k}(v)$ for $k = 2$

```

Int_f1v_Gau_k
= ( (sqrt(M_PI) / 4.0) *
    (v_0 * v_0 * v_0) *
    erf(v_max / v_0)
  - v_max *
    ( v_0 * v_0 / 2.0
    - v_max * v_max / 3.0
    - 2.0 * v_0 * v_0 ) *
    exp(-(v_max * v_max) / (v_0 * v_0))
  - sqrt(2.0 * M_PI) *
    (v_0 * v_0 * v_0) *
    erf(v_max / v_0 / sqrt(2.0)) *
    exp(-(v_max * v_max) / (2.0 * v_0 * v_0)) ) *
v_U;

```

For $k = 3$, we have

$$\begin{aligned}
& \int_0^{v_{\max}} f_{1,\text{Gau},k=3}(v) dv \\
&= \frac{1}{N_{f,k=3}} \int_0^{v_{\max}} v^2 \left(e^{-v^2/3v_0^2} - e^{-v_{\max}^2/3v_0^2} \right)^3 dv \\
&= \frac{1}{N_{f,k=3}} \left\{ \frac{\sqrt{\pi}}{4} v_0^3 \operatorname{erf}\left(\frac{v_{\max}}{v_0}\right) - v_{\max} \left(\frac{v_0^2}{2} + \frac{v_{\max}^2}{3} + \frac{9v_0^2}{4} \right) e^{-v_{\max}^2/v_0^2} \right. \\
&\quad \left. - v_0^3 \left[\frac{9\sqrt{6\pi}}{16} \operatorname{erf}\left(\frac{\sqrt{2}v_{\max}}{\sqrt{3}v_0}\right) e^{-v_{\max}^2/3v_0^2} - \frac{9\sqrt{3\pi}}{4} \operatorname{erf}\left(\frac{v_{\max}}{\sqrt{3}v_0}\right) e^{-2v_{\max}^2/3v_0^2} \right] \right\} \\
&= 1,
\end{aligned} \tag{A4}$$

and then can define N_f by

Code A13: Normalization constant N_f of $f_{1,\text{Gau},k}(v)$ for $k = 3$

```

Int_f1v_Gau_k
= ( (sqrt(M_PI) / 4.0) *
    (v_0 * v_0 * v_0) *
    erf(v_max / v_0)
  - v_max *
    ( v_0 * v_0 / 2.0
      + v_max * v_max / 3.0
      + (9.0 / 4.0) * v_0 * v_0 )
    exp(-(v_max * v_max) / (v_0 * v_0) )
  + (v_0 * v_0 * v_0) *
    (- sqrt(6.0 * M_PI) * (9.0 / 16.0)
      erf( v_max / v_0 * (sqrt(6.0) / 3.0) )
      exp(-(v_max * v_max) / (3.0 * v_0 * v_0) )
    + sqrt(3.0 * M_PI) * (9.0 / 4.0)
      erf( v_max / v_0 / sqrt(3.0) )
      exp(-(2.0 * v_max * v_max) / (3.0 * v_0 * v_0) ) ) ) *
v_U;

```

For $k = 4$, we have

$$\begin{aligned}
& \int_0^{v_{\max}} f_{1,\text{Gau},k=4}(v) dv \\
&= \frac{1}{N_{f,k=4}} \int_0^{v_{\max}} v^2 \left(e^{-v^2/4v_0^2} - e^{-v_{\max}^2/4v_0^2} \right)^4 dv \\
&= \frac{1}{N_{f,k=4}} \left\{ \frac{\sqrt{\pi}}{4} v_0^3 \operatorname{erf} \left(\frac{v_{\max}}{v_0} \right) - v_{\max} \left(\frac{v_0^2}{2} - \frac{v_{\max}^2}{3} - \frac{14v_0^2}{3} \right) e^{-v_{\max}^2/v_0^2} \right. \\
&\quad \left. - v_0^3 \left[\frac{8\sqrt{3\pi}}{9} \operatorname{erf} \left(\frac{\sqrt{3} v_{\max}}{2v_0} \right) e^{-v_{\max}^2/4v_0^2} - 3\sqrt{2\pi} \operatorname{erf} \left(\frac{v_{\max}}{\sqrt{2} v_0} \right) e^{-v_{\max}^2/2v_0^2} \right. \right. \\
&\quad \left. \left. + 8\sqrt{\pi} \operatorname{erf} \left(\frac{v_{\max}}{2v_0} \right) e^{-3v_{\max}^2/4v_0^2} \right] \right\} \\
&= 1,
\end{aligned} \tag{A5}$$

and then can define N_f by

Code A14: Normalization constant N_f of $f_{1,\text{Gau},k}(v)$ for $k = 4$

```

Int_f1v_Gau_k
= ( (sqrt(M_PI) / 4.0) *
    (v_0 * v_0 * v_0) *
    erf(v_max / v_0)
    - v_max *
    ( v_0 * v_0 / 2.0
      - v_max * v_max / 3.0
      - (14.0 / 3.0) * v_0 * v_0 )
    exp(-(v_max * v_max) / (v_0 * v_0) )
    + (v_0 * v_0 * v_0)
    (- sqrt(3.0 * M_PI) * (8.0 / 9.0)
      erf( v_max / v_0 * (sqrt(3.0) / 2.0) )
      exp(-(v_max * v_max) / (4.0 * v_0 * v_0) )
    + sqrt(2.0 * M_PI) * 3.0
      erf( v_max / v_0 / sqrt(2.0) )
      exp(-(v_max * v_max) / (2.0 * v_0 * v_0) )
    - sqrt(M_PI) * 8.0
      erf( v_max / v_0 / 2.0 )
      exp(-(3.0 * v_max * v_max) / (4.0 * v_0 * v_0) ) ) ) *
v_U;

```

Now we can define the modified Maxwellian velocity distribution given in Eq. (2) for $k = 1, 2, 3$, and 4 as

Code A15: Modified Maxwellian velocity distribution $f_{1,\text{Gau},k}(v)$

```

double f1v_Gau_k(double vv)
{
    return
        ( (vv * vv) *
          pow( exp(-(vv * vv) / (k_f1v * v_0 * v_0) )
              - exp(-(v_max * v_max) / (k_f1v * v_0 * v_0) ),
              k_f1v ) ) /
        Int_f1v_Gau_k;
}

```

Meanwhile, for predicting the WIMP–signal spectrum one needs also that, for $k = 1$,

$$\begin{aligned}
 \int \left[\frac{f_{1,\text{Gau},k=1}(v)}{v} \right] dv &= \frac{1}{N_{f,k=1}} \int v \left(e^{-v^2/v_0^2} - e^{-v_{\text{max}}^2/v_0^2} \right) dv \\
 &= \frac{1}{N_{f,k=1}} \left(-\frac{v_0^2}{2} e^{-v^2/v_0^2} - \frac{v^2}{2} e^{-v_{\text{max}}^2/v_0^2} \right), \tag{A6}
 \end{aligned}$$

to define

Code A16: Integral over the modified Maxwellian velocity distribution $\int [f_{1,\text{Gau},k}(v)/v] dv$ for $k = 1$

```
double Int_f1v_v_Gau_k(double vv)
{
    return
        (- (v_0 * v_0) / 2.0 *
         exp(-(vv * vv) / (v_0 * v_0)) -
         (vv * vv) / 2.0 *
         exp(-(v_max * v_max) / (v_0 * v_0)) ) /
        Int_f1v_v_Gau_k;
}
```

For $k = 2$, one has

$$\begin{aligned} \int \left[\frac{f_{1,\text{Gau},k=2}(v)}{v} \right] dv &= \frac{1}{N_{f,k=2}} \int v \left(e^{-v^2/2v_0^2} - e^{-v_{\text{max}}^2/2v_0^2} \right)^2 dv \\ &= \frac{1}{N_{f,k=2}} \left(-\frac{v_0^2}{2} e^{-v^2/v_0^2} + \frac{v^2}{2} e^{-v_{\text{max}}^2/v_0^2} + 2v_0^2 e^{-v_{\text{max}}^2/2v_0^2} e^{-v^2/2v_0^2} \right), \quad (\text{A7}) \end{aligned}$$

and can define

Code A17: Integral over the modified Maxwellian velocity distribution $\int [f_{1,\text{Gau},k}(v)/v] dv$ for $k = 2$

```
double Int_f1v_v_Gau_k(double vv)
{
    return
        (- (v_0 * v_0) / 2.0 *
         exp(-(vv * vv) / (v_0 * v_0)) +
         (vv * vv) / 2.0 *
         exp(-(v_max * v_max) / (v_0 * v_0)) +
         2.0 *
         (v_0 * v_0) *
         exp(-(v_max * v_max) / (2.0 * v_0 * v_0)) *
         exp(-(vv * vv) / (2.0 * v_0 * v_0)) ) /
        Int_f1v_v_Gau_k;
}
```

For $k = 3$, one has

$$\begin{aligned} \int \left[\frac{f_{1,\text{Gau},k=3}(v)}{v} \right] dv &= \frac{1}{N_{f,k=3}} \int v \left(e^{-v^2/3v_0^2} - e^{-v_{\text{max}}^2/3v_0^2} \right)^3 dv \\ &= \frac{1}{N_{f,k=3}} \left[-\frac{v_0^2}{2} e^{-v^2/v_0^2} - \frac{v^2}{2} e^{-v_{\text{max}}^2/v_0^2} \right. \\ &\quad \left. + \frac{9}{4} v_0^2 \left(e^{-v_{\text{max}}^2/3v_0^2} e^{-2v^2/3v_0^2} - 2e^{-2v_{\text{max}}^2/3v_0^2} e^{-v^2/3v_0^2} \right) \right], \quad (\text{A8}) \end{aligned}$$

and can define

Code A18: Integral over the modified Maxwellian velocity distribution $\int [f_{1,\text{Gau},k}(v)/v] dv$ for $k = 3$

```
double Int_f1v_v_Gau_k(double vv)
{
    return
        (- (v_0 * v_0) / 2.0 *
         exp(-(vv * vv) / (v_0 * v_0)) -
         (vv * vv) / 2.0 *
         exp(-(v_max * v_max) / (v_0 * v_0)) +
         (v_0 * v_0) *
         ( (9.0 / 4.0) *
          exp(-(v_max * v_max) / (3.0 * v_0 * v_0)) *
          exp(-(2.0 * vv * vv) / (3.0 * v_0 * v_0)) -
          (9.0 / 2.0) *
          exp(-(2.0 * v_max * v_max) / (3.0 * v_0 * v_0)) *
          exp(-(vv * vv) / (3.0 * v_0 * v_0)) ) ) ) /
    Int_f1v_Gau_k;
```

For $k = 4$, one has

$$\begin{aligned}
 & \int \left[\frac{f_{1,\text{Gau},k=4}(v)}{v} \right] dv \\
 &= \frac{1}{N_{f,k=4}} \int v \left(e^{-v^2/4v_0^2} - e^{-v_{\text{max}}^2/4v_0^2} \right)^4 dv \\
 &= \frac{1}{N_{f,k=4}} \left[-\frac{v_0^2}{2} e^{-v^2/v_0^2} + \frac{v^2}{2} e^{-v_{\text{max}}^2/v_0^2} \right. \\
 & \quad \left. + v_0^2 \left(\frac{8}{3} e^{-v_{\text{max}}^2/4v_0^2} e^{-3v^2/4v_0^2} - 6 e^{-v_{\text{max}}^2/2v_0^2} e^{-v^2/2v_0^2} + 8 e^{-3v_{\text{max}}^2/4v_0^2} e^{-v^2/4v_0^2} \right) \right], \text{(A9)}
 \end{aligned}$$

and can define

Code A19: Integral over the modified Maxwellian velocity distribution $\int [f_{1,\text{Gau},k}(v)/v] dv$ for $k = 4$

```
double Int_f1v_v_Gau_k(double vv)
{
    return
        (- (v_0 * v_0) / 2.0 *
         exp(-(vv * vv) / (v_0 * v_0)) +
         (vv * vv) / 2.0 *
         exp(-(v_max * v_max) / (v_0 * v_0)) +
         (v_0 * v_0) *
         ( (8.0 / 3.0) *
          exp(-(v_max * v_max) / (4.0 * v_0 * v_0)) *
          exp(-(3.0 * vv * vv) / (4.0 * v_0 * v_0)) -
          6.0 *
          exp(-(v_max * v_max) / (2.0 * v_0 * v_0)) *
          exp(-(vv * vv) / (2.0 * v_0 * v_0)) +
          8.0 *
          exp(-(3.0 * v_max * v_max) / (4.0 * v_0 * v_0)) *
          exp(-(vv * vv) / (4.0 * v_0 * v_0)) ) ) ) /
    Int_f1v_Gau_k;
```


Hence, we can define the required integral

$$\int_{\alpha\sqrt{Q}}^{v_{\max}} \left[\frac{f_{1,\text{Gau},k}(v)}{v} \right] dv \quad (\text{A10})$$

for $k = 1, 2, 3$ and 4 as

Code A20: Integral over the modified Maxwellian velocity distribution $\int_{\alpha\sqrt{Q}}^{v_{\max}} [f_{1,\text{Gau},k}(v)/v] dv$

```
double Intf1v_v_Gau_k(double mchi, int A, double QQ)
{
    return
        Int_f1v_v_Gau_k(v_max)
        - Int_f1v_v_Gau_k(alpha(mchi, A) * sqrt(QQ) / v_U);
}
```

On the other hand, while for drawing the output generating velocity distribution function, we need to define, for $k = 1, 2, 3$ and 4,

Code A21: Modified Maxwellian velocity distribution $f_{1,\text{Gau},k}(v)$ (for Gnuplot)

```
f1v_Gau_k(x)
= (x * x) *
  ( exp(-(x * x) / (k_f1v * v_0 * v_0))
    - exp(-(v_max * v_max) / (k_f1v * v_0 * v_0)) ) ** k_f1v /
Int_f1v_Gau_k
```

where the normalization constant `Int_f1v_Gau_k` will be estimated directly by Codes A11 to A14, for drawing the output generating WIMP-signal spectrum, we need

Code A22: Integral over the modified Maxwellian velocity distribution $\int [f_{1,\text{Gau},k}(v)/v] dv$ for $k = 1$ (for Gnuplot)

```
Int_f1v_v_Gau_k(x)
= (- (v_0 * v_0) / 2.0
   * exp(-(x * x) / (v_0 * v_0))
   - (x * x) / 2.0
   * exp(-(v_max * v_max) / (v_0 * v_0)) ) /
Int_f1v_Gau_k
```

Code A23: Integral over the modified Maxwellian velocity distribution $\int [f_{1,\text{Gau},k}(v)/v] dv$ for $k = 2$ (for Gnuplot)

```
Int_f1v_v_Gau_k(x)
= (- (v_0 * v_0) / 2.0
   * exp(-(x * x) / (v_0 * v_0))
   + (x * x) / 2.0
   * exp(-(v_max * v_max) / (v_0 * v_0))
   + 2.0
   * (v_0 * v_0)
   * exp(-(v_max * v_max) / (2.0 * v_0 * v_0))
   * exp(-(x * x) / (2.0 * v_0 * v_0)) ) /
Int_f1v_Gau_k
```

Code A24: Integral over the modified Maxwellian velocity distribution $\int [f_{1,\text{Gau},k}(v)/v] dv$ for $k = 3$ (for Gnuplot)

```

Int_f1v_v_Gau_k(x)
=  (- (v_0 * v_0) / 2.0
    exp(-(x      * x      ) / (v_0 * v_0) )
    - (x      * x      ) / 2.0
    exp(-(v_max * v_max) / (v_0 * v_0) )
    + (v_0 * v_0)
    ( (9.0 / 4.0)
      exp(-(      v_max * v_max) / (3.0 * v_0 * v_0) ) *
      exp(-(2.0 * x      * x      ) / (3.0 * v_0 * v_0) )
      - (9.0 / 2.0)
      exp(-(2.0 * v_max * v_max) / (3.0 * v_0 * v_0) ) *
      exp(-(      x      * x      ) / (3.0 * v_0 * v_0) ) ) ) /
Int_f1v_Gau_k

```

Code A25: Integral over the modified Maxwellian velocity distribution $\int [f_{1,\text{Gau},k}(v)/v] dv$ for $k = 4$ (for Gnuplot)

```

Int_f1v_v_Gau_k(x)
=  (- (v_0 * v_0) / 2.0
    exp(-(x      * x      ) / (v_0 * v_0) )
    + (x      * x      ) / 2.0
    exp(-(v_max * v_max) / (v_0 * v_0) )
    + (v_0 * v_0)
    ( (8.0 / 3.0)
      exp(-(      v_max * v_max) / (4.0 * v_0 * v_0) ) *
      exp(-(3.0 * x      * x      ) / (4.0 * v_0 * v_0) )
      - 6.0
      exp(-(      v_max * v_max) / (2.0 * v_0 * v_0) ) *
      exp(-(      x      * x      ) / (2.0 * v_0 * v_0) )
      + 8.0
      exp(-(3.0 * v_max * v_max) / (4.0 * v_0 * v_0) ) *
      exp(-(      x      * x      ) / (4.0 * v_0 * v_0) ) ) ) /
Int_f1v_Gau_k

```

for finally defining

Code A26: Integral over the modified Maxwellian velocity distribution $\int_{\alpha\sqrt{Q}}^{v_{\max}} [f_{1,\text{Gau},k}(v)/v] dv$ (for Gnuplot)

```

c      = 1.0
m_U    = 1e6 / (c * c)
v_U    = c / (2.9979246 * 1e5)

m_N    = (0.938272 * m_U) * AX * 0.99
m_rN   = (m_chi * m_U) * m_N / (m_chi * m_U + m_N)

alpha  = sqrt(m_N / (2.0 * m_rN * m_rN))

Intf1v_v_Gau_k(x)
=  Int_f1v_v_Gau_k(v_max)
    - Int_f1v_v_Gau_k(alpha * sqrt(x) / v_U)

```

Remind that firstly, “\” (backslash) must be used in order to let the including of the definition(s) given in this file into the other intrinsic commands correctly. Secondly, all flexible-kept parameters: `k_f1v` (stands for the power index k of the modified Maxwellian velocity distribution), `v_0`, `v_max` as well as `AX` and `m_chi` will be read directly from the users’ initial (simulation/data analysis) setup set earlier on the website.

B.1.3 Shifted Maxwellian velocity distribution $f_{1,\text{sh}}(v)$

According to Eqs. (3) and (4), we define

Code A27: Shifted Maxwellian velocity distribution $f_{1,\text{sh}}(v)$

```
double f1v_sh(double vv, double tt)
{
    return
        (1.0 / sqrt(M_PI)) *
        (vv / v_0 / v_e(tt)) / v_U *
        ( exp(-(vv - v_e(tt)) * (vv - v_e(tt)) / (v_0 * v_0))
          - exp(-(vv + v_e(tt)) * (vv + v_e(tt)) / (v_0 * v_0)) );
}
```

and

Code A28: Time-dependent Earth’s velocity in the Galactic frame $v_e(t)$

```
double v_e(double tt)
{
    return v_0 * ( 1.05 + 0.07 * cos(omega * (tt - t_p)) );
}
```

Then, since

$$\int_{\alpha\sqrt{Q}}^{\infty} \left[\frac{f_{1,\text{sh}}(v)}{v} \right] dv = \frac{1}{2v_e} \left[\text{erf} \left(\frac{\alpha\sqrt{Q} + v_e}{v_0} \right) - \text{erf} \left(\frac{\alpha\sqrt{Q} - v_e}{v_0} \right) \right], \quad (\text{A11})$$

we can define

Code A29: Integral over the shifted Maxwellian velocity distribution $\int_{\alpha\sqrt{Q}}^{\infty} [f_{1,\text{sh}}(v)/v] dv$

```
double Intf1v_v_sh(double mchi, int A, double QQ, double tt)
{
    return
        (1.0 / 2.0 / (v_e(tt) * v_U)) *
        ( erf( (alpha(mchi, A) * sqrt(QQ) + (v_e(tt) * v_U)) /
              (v_0 * v_U) )
          - erf( (alpha(mchi, A) * sqrt(QQ) - (v_e(tt) * v_U)) /
              (v_0 * v_U) ) );
}
```

On the other hand, for drawing output plots, we have

Code A30: Shifted Maxwellian velocity distribution $f_{1,\text{sh}}(v)$ (for Gnuplot)

```
M_PI    = 3.141593
omega   = 2.0 * M_PI / 365.0

v_e = v_0 * ( 1.05 + 0.07 * cos(omega * (t_expt - t_p)) )

f1v_sh(x)                                     \
= (1.0 / sqrt(M_PI)) *                       \
  (x / v_0 / v_e) *                           \
  ( exp(-(x - v_e) * (x - v_e) / (v_0 * v_0)) \
    - exp(-(x + v_e) * (x + v_e) / (v_0 * v_0)) )
```

and

Code A31: Integral over the shifted Maxwellian velocity distribution $\int_{\alpha\sqrt{Q}}^{\infty} [f_{1,\text{sh}}(v)/v] dv$ (for Gnuplot)

```
c      = 1.0
m_U    = 1e6 / (c * c)
v_U    = c / (2.9979246 * 1e5)

m_N    = (0.938272 * m_U) * AX * 0.99
m_rN   = (m_chi * m_U) * m_N / (m_chi * m_U + m_N)

alpha  = sqrt(m_N / (2.0 * m_rN * m_rN))

Intf1v_v_sh(x)                               \
= (1.0 / 2.0 / (v_e * v_U)) *                 \
  ( erf( ( alpha * sqrt(x) + (v_e * v_U) ) / (v_0 * v_U) ) \
    - erf( ( alpha * sqrt(x) - (v_e * v_U) ) / (v_0 * v_U) ) )
```

Remind here that, firstly, “\” (backslash) must be used in order to let the including of the definition given in this file into the other intrinsic commands correctly. Secondly, all flexible-kept parameters: t_p , t_{expt} , v_0 as well as AX and m_{chi} will be read directly from the users’ initial (simulation/data analysis) setup set earlier on the website.

B.2 Elastic nuclear form factors

In this subsection, we give the codes for intrinsically defined elastic nuclear form factors given in Sec. 3.4 in the AMIDAS package.

B.2.1 Exponential form factor $F_{\text{ex}}(Q)$

For the exponential form factor $F_{\text{ex}}(Q)$, the nuclear radius $R_0(A)$ and the nuclear coherence energy $Q_0(A)$ have been defined as functions of the atomic mass number A :

Code A32: Nuclear radius $R_0(A)$

```
double R_0(int A)
{
  return (0.3 + 0.91 * cbrt(m_N(A) / m_U)) * fm_U;
}
```

and

Code A33: Nuclear coherence energy $Q_0(A)$

```
double Q_0(int A)
{
    return 1.5 / (m_N(A) * R_0(A) * R_0(A));
}
```

Then we can define

Code A34: Squared exponential form factor $F_{\text{ex}}^2(A, Q)$

```
double FQ_ex(int A, double QQ)
{
    return exp(-QQ / Q_0(A));
}
```

and

Code A35: Derivative of the squared exponential form factor $dF_{\text{ex}}^2(A, Q)/dQ$

```
double dFQdQ_ex(int A, double QQ)
{
    return -(1.0 / Q_0(A)) * FQ_ex(A, QQ);
}
```

since

$$\frac{dF_{\text{ex}}^2(Q)}{dQ} = \frac{d}{dQ} \left(e^{-Q/Q_0} \right) = -\frac{1}{Q_0} e^{-Q/Q_0}. \quad (\text{A12})$$

On the other hand, for drawing the output generating WIMP–signal recoil spectrum, we need

Code A36: Squared exponential form factor $F_{\text{ex}}^2(Q)$ (for Gnuplot)

```
c      = 1.0
m_U    = 1e6 / (c * c)
fm_U   = c / (0.197327 * 1e6)

m_N    = (0.938272 * m_U) * AX * 0.99
m_rN   = (m_chi * m_U) * m_N / (m_chi * m_U + m_N)

alpha  = sqrt(m_N / (2.0 * m_rN * m_rN))

R_0    = ( 0.3 + 0.91 * (m_N / m_U) ** (1.0 / 3.0) ) * fm_U
Q_0    = 1.5 / (m_N * R_0 * R_0)

      FQ_ex(x)      \
=   exp(-x / Q_0)
```

Remind here that, firstly, “\” (backslash) must be used in order to let the including of the definition given in this file into the other intrinsic commands correctly. Secondly, two flexible–kept parameters: `AX` and `m_chi` will be read directly from the users’ initial (simulation/data analysis) setup set earlier on the website.

B.2.2 Woods–Saxon form factor $F_{\text{WS}}(Q)$

For the Woods–Saxon form factor $F_{\text{WS}}(Q)$, the nuclear skin thickness s , the radius $R_A(A)$, and the effective nuclear radius $R_1(A)$ are defined as

Code A37: Nuclear skin thickness s

```
double ss = fm_U;
```

and

Code A38: $R_A(A)$

```
double R_A(int A)
{
    return 1.2 * cbrt(A) * fm_U;
}
```

Then we have

Code A39: Effective nuclear radius $R_1(A)$

```
double R_1(int A)
{
    return sqrt(R_A(A) * R_A(A) - 5.0 * ss * ss);
}
```

Meanwhile, the transferred 3-momentum q given in Eq. (12) has been give as a function of the atomic mass number A and recoil energy Q :

Code A40: Transferred 3-momentum $q(A, Q)$

```
double qq(int A, double QQ)
{
    return sqrt(2.0 * m_N(A) * QQ);
}
```

Then we can define

Code A41: Squared Woods–Saxon form factor $F_{\text{WS}}^2(A, Q)$

```
double FQ_WS(int A, double QQ)
{
    if (QQ == 0.0)
    {
        return 1.0;
    }

    else
    {
        return
            ( 3.0 * sphBesselj(1, qq(A, QQ) * R_1(A)) / (qq(A, QQ) * R_1(A)) ) *
            ( 3.0 * sphBesselj(1, qq(A, QQ) * R_1(A)) / (qq(A, QQ) * R_1(A)) ) *
            exp(-(qq(A, QQ) * ss) * (qq(A, QQ) * ss));
    }
}
```

For defining the derivative of the squared Woods–Saxon form factor, $dF_{\text{WS}}^2(Q)/dQ$, first, we find that

$$\begin{aligned}
\frac{dF_{\text{WS}}^2(Q)}{dQ} &= \frac{d}{dQ} \left\{ \left[\frac{3j_1(qR_1)}{qR_1} \right]^2 e^{-(qs)^2} \right\} \\
&= 2 \left[\frac{3j_1(qR_1)}{qR_1} \right] e^{-(qs)^2} \left[\frac{3j_1'(qR_1)(q'R_1)(qR_1) - 3j_1(qR_1)(q'R_1)}{(qR_1)^2} \right] \\
&\quad + \left[\frac{3j_1(qR_1)}{qR_1} \right]^2 e^{-(qs)^2} \left[(-2qs^2) q' \right] \\
&= 2 \left[\frac{3j_1(qR_1)}{qR_1} \right]^2 e^{-(qs)^2} \left[\frac{j_1'(qR_1)(qR_1) - j_1(qR_1)}{j_1(qR_1)q} - qs^2 \right] \left(\frac{dq}{dQ} \right) \\
&= \left[\frac{j_1'(qR_1)R_1}{j_1(qR_1)} - \frac{1}{q} - qs^2 \right] \left(\frac{2m_N}{q} \right) F_{\text{WS}}^2(Q), \tag{A13}
\end{aligned}$$

where, from the expression (12) for q , we have

$$\frac{dq}{dQ} = \sqrt{2m_N} \cdot \frac{1}{2\sqrt{Q}} = \sqrt{2m_N} \cdot \frac{\sqrt{2m_N}}{2q} = \frac{m_N}{q}. \tag{A14}$$

The expression (A13) can be used for $Q > 0$. However, for the limit case $Q = q = 0$, since [42]

$$j_1(x) = \sqrt{\frac{\pi}{2x}} J_{3/2}(x) = \frac{1}{x} \left(\frac{\sin x}{x} - \cos x \right), \tag{A15}$$

we can get

$$j_1'(x) = \frac{(x^2 - 2) \sin x + 2x \cos x}{x^3}. \tag{A16}$$

Then, by letting $x = qR_1$, the first two terms in the bracket of the right-hand side of Eq. (A13) (multiply by $1/q$) can be written and found that

$$\frac{j_1'(x)}{j_1(x)} \left(\frac{R_1^2}{x} \right) - \frac{R_1^2}{x^2} = \left[\frac{(x^2 - 2) \sin x + 2x \cos x}{x^2 (\sin x - x \cos x)} - \frac{1}{x^2} \right] R_1^2 \rightarrow -\frac{R_1^2}{5}, \tag{A17}$$

as $x \rightarrow 0$. Hence, we can obtain that, for the limit case $Q = q = 0$,

$$\left. \frac{dF_{\text{WS}}^2(Q)}{dQ} \right|_{Q=0} = \left(-\frac{R_1^2}{5} - s^2 \right) (2m_N) = -\left(\frac{2}{5} \right) R_A^2 m_N, \tag{A18}$$

where we use the expression (13) for R_1 and

$$F_{\text{WS}}^2(Q = 0) = 1. \tag{A19}$$

Now, we can define the derivative of the squared Woods–Saxon form factor as

Code A42: Derivative of the squared Woods–Saxon form factor $dF_{\text{WS}}^2(A, Q)/dQ$

```
double dFQdQ_WS(int A, double QQ)
{
  if (QQ == 0.0)
  {
    return -0.4 * (R_A(A) * R_A(A)) * m_N(A);
  }

  else
  {
    return
      ( dsphBesselj(1, qq(A, QQ) * R_1(A)) * R_1(A) /
        sphBesselj(1, qq(A, QQ) * R_1(A))
        - 1.0 / qq(A, QQ)
        - (qq(A, QQ) * ss * ss) ) *
      FQ_WS(A, QQ) * ((2.0 * m_N(A)) / qq(A, QQ));
  }
}
```

Moreover, the needed spherical Bessel functions

$$j_n(x) = \sqrt{\frac{\pi}{2x}} J_{n+1/2}(x), \quad (\text{A20})$$

and their derivatives

$$j'_n(x) = \sqrt{\frac{\pi}{2x}} \left[-\frac{1}{2x} J_{n+1/2}(x) + J'_{n+1/2}(x) \right], \quad (\text{A21})$$

have also been defined in an intrinsic package of the **AMIDAS** code. We define at first the half-integer Bessel functions $J_{n+1/2}(x)$, for $n = 0, \pm 1, \pm 2, \dots$, by

Code A43: Bessel functions $J_{n/2}(x)$ for $n = \pm 1, \pm 3, \pm 5, \dots$

```
double BesselJ(int n, int m, double x)
{
  if (m == 2)
  {
    switch (n)
    {
      case 1:
        return
          sqrt(2.0 / M_PI / x) * sin(x);
        break;

      case -1:
        return
          sqrt(2.0 / M_PI / x) * cos(x);
        break;

      case 3:
        return
          sqrt(2.0 / M_PI / x) * (sin(x) / x - cos(x));
        break;
    }
  }
}
```



```

case -3:
    return
    sqrt(2.0 / M_PI / x) * (cos(x) / x + sin(x)) * (-1);
    break;

case 5:
    return
    sqrt(2.0 / M_PI / x) * ((3.0 / (x * x) - 1) * sin(x) - (3.0 / x) * cos(x));
    break;

case -5:
    return
    sqrt(2.0 / M_PI / x) * ((3.0 / (x * x) - 1) * cos(x) + (3.0 / x) * sin(x));
    break;

default:
    if (n > 5 && n % 2 == 1)
    {
        return
            ((n - 2) / x) * BesselJ(n - 2, 2, x) - BesselJ(n - 4, 2, x);
    }

    else
    if (n < -5 && -n % 2 == 1)
    {
        return
            ((n + 2) / x) * BesselJ(n + 2, 2, x) - BesselJ(n + 4, 2, x);
    }
    }
}
}

```

Here for the case $n \geq 3$ or $n \leq -3$ (i.e. $\pm n \geq 3$), we use the following relation [42]:

$$J_{n\pm 1}(x) = \left(\frac{2n}{x}\right) J_n(x) - J_{n\mp 1}(x). \quad (\text{A22})$$

Then one can derive that

$$\begin{aligned}
 J_{m/2}(x) &= \left[\frac{2(m/2 \mp 1)}{x} \right] J_{m/2 \mp 1}(x) - J_{m/2 \mp 2}(x) \\
 &= \left(\frac{m \mp 2}{x} \right) J_{(m \mp 2)/2}(x) - J_{(m \mp 4)/2}(x).
 \end{aligned} \quad (\text{A23})$$

For the derivatives of the Bessel functions, since [42]

$$\begin{aligned}
 J'_n(x) &= \frac{1}{2} [J_{n-1}(x) - J_{n+1}(x)] \\
 &= -\left(\frac{n}{x}\right) J_n(x) + J_{n-1}(x) \\
 &= \left(\frac{n}{x}\right) J_n(x) - J_{n+1}(x),
 \end{aligned} \quad (\text{A24})$$

we can find that

$$J'_{m/2}(x) = \mp \left(\frac{m}{2x} \right) J_{m/2}(x) \pm J_{(m \mp 2)/2}(x). \quad (\text{A25})$$

Therefore, according to Eq. (A20), we can first define

Code A44: Spherical Bessel functions $j_n(x)$ for $n = 0, 1, 2, \dots$

```
double sphBesselj(int n, double x)
{
    return sqrt(M_PI / 2.0 / x) * BesselJ(2 * n + 1, 2, x);
}
```

Meanwhile, from Eqs. (A21) and (A24), we can obtain that

$$\begin{aligned} j'_n(x) &= \sqrt{\frac{\pi}{2x}} \left[-\frac{1}{2x} J_{n+1/2}(x) + J'_{n+1/2}(x) \right] \\ &= \sqrt{\frac{\pi}{2x}} \left\{ -\frac{1}{2x} J_{n+1/2}(x) + \frac{1}{2} [J_{n-1/2}(x) - J_{n+3/2}(x)] \right\} \\ &= \sqrt{\frac{\pi}{2x}} \cdot \frac{1}{2} \left\{ -\frac{1}{x} J_{n+1/2}(x) + J_{n-1/2}(x) - \left[\left(\frac{2n+1}{x} \right) J_{n+1/2}(x) - J_{n-1/2}(x) \right] \right\} \\ &= \sqrt{\frac{\pi}{2x}} \left[-\left(\frac{n+1}{x} \right) J_{n+1/2}(x) + J_{n-1/2}(x) \right], \end{aligned} \quad (\text{A26})$$

and thus define

Code A45: Derivatives of the Spherical Bessel functions $j'_n(x)$ for $n = 0, 1, 2, \dots$

```
double dsphBesselj(int n, double x)
{
    return
        sqrt(M_PI / 2.0 / x) *
        (- (n + 1) * BesselJ(2 * n + 1, 2, x) / x
         + BesselJ(2 * n - 1, 2, x));
}
```

Finally, for drawing the output generating WIMP–signal recoil spectrum, we need

Code A46: Squared Woods–Saxon form factor $F_{\text{WS}}^2(Q)$ (for Gnuplot)

```

c      = 1.0
m_U    = 1e6 / (c * c)
fm_U   = c / (0.197327 * 1e6)

m_N    = (0.938272 * m_U) * AX * 0.99

ss     = fm_U
R_A    = 1.2 * (AX ** (1.0 / 3.0)) * fm_U
R_1    = sqrt(R_A * R_A - 5.0 * ss * ss)

      qq(x) \
= sqrt(2.0 * m_N * x)

      sphBesselj_1(x) \
= ( sin(qq(x) * R_1) / (qq(x) * R_1) - cos(qq(x) * R_1) ) / \
  (qq(x) * R_1)

      FQ_WS(x) \
= ( 3.0 * sphBesselj_1(x) / (qq(x) * R_1) ) * \
  ( 3.0 * sphBesselj_1(x) / (qq(x) * R_1) ) * \
  exp(-(qq(x) * ss) * (qq(x) * ss))

```

Remind here that, firstly, “\” (backslash) must be used in order to let the including of the definition given in this file into the other intrinsic commands correctly. Secondly, the flexible-kept parameter: AX will be read directly from the users’ initial (simulation/data analysis) setup set earlier on the website.

B.2.3 Modified Woods–Saxon form factor $F_{\text{WS,Eder}}(Q)$

First, the modified radius R_A given in Eq. (16) has been defined as

Code A47: Modified $R_A(A)$

```

double R_A_Eder(int A)
{
    return (1.15 * cbrt(A) + 0.39) * fm_U;
}

```

Then the effective nuclear radius R_1 has to be redefined by

Code A48: Modified effective nuclear radius $R_1(A)$

```

double R_1_Eder(int A)
{
    return sqrt(R_A_Eder(A) * R_A_Eder(A) - 5.0 * ss * ss);
}

```

Therefore, similar to Codes A41 and A42 for the (derivative of the) Woods–Saxon form factor $F_{\text{WS}}^2(Q)$, we can define

Code A49: Squared modified Woods–Saxon form factor $F_{\text{WS,Eder}}^2(A, Q)$

```
double FQ_WS_Eder(int A, double QQ)
{
    if (QQ == 0.0)
    {
        return 1.0;
    }

    else
    {
        return
            ( 3.0 * sphBesselj(1, qq(A, QQ) * R_1_Eder(A)) / (qq(A, QQ) * R_1_Eder(A)) ) *
            ( 3.0 * sphBesselj(1, qq(A, QQ) * R_1_Eder(A)) / (qq(A, QQ) * R_1_Eder(A)) ) *
            exp(-(qq(A, QQ) * ss) * (qq(A, QQ) * ss));
    }
}
```

and

Code A50: Derivative of the squared modified Woods–Saxon form factor $dF_{\text{WS,Eder}}^2(A, Q)/dQ$

```
double dFQdQ_WS_Eder(int A, double QQ)
{
    if (QQ == 0.0)
    {
        return -0.4 * (R_A_Eder(A) * R_A_Eder(A)) * m_N(A);
    }

    else
    {
        return
            ( dsphBesselj(1, qq(A, QQ) * R_1_Eder(A)) * R_1_Eder(A) /
              sphBesselj(1, qq(A, QQ) * R_1_Eder(A))
              - 1.0 / qq(A, QQ)
              - (qq(A, QQ) * ss * ss) ) *
            FQ_WS_Eder(A, QQ) * ((2.0 * m_N(A)) / qq(A, QQ));
    }
}
```

On the other hand, for drawing the output generating WIMP–signal recoil spectrum, we need

Code A51: Squared modified Woods–Saxon form factor $F_{\text{WS,Eder}}^2(Q)$ (for Gnuplot)

```

c      = 1.0
m_U    = 1e6 / (c * c)
fm_U   = c / (0.197327 * 1e6)

m_N    = (0.938272 * m_U) * AX * 0.99

ss      = fm_U
R_A_Eder = ( 1.15 * (AX ** (1.0 / 3.0)) + 0.39 ) * fm_U
R_1_Eder = sqrt(R_A_Eder * R_A_Eder - 5.0 * ss * ss)

      qq(x) \
= sqrt(2.0 * m_N * x)

      sphBesselj_1(x) \
= ( sin(qq(x) * R_1_Eder) / (qq(x) * R_1_Eder) - cos(qq(x) * R_1_Eder) ) / \
  (qq(x) * R_1_Eder)

      FQ_WS_Eder(x) \
= ( 3.0 * sphBesselj_1(x) / (qq(x) * R_1_Eder) ) * \
  ( 3.0 * sphBesselj_1(x) / (qq(x) * R_1_Eder) ) * \
  exp(-(qq(x) * ss) * (qq(x) * ss))

```

Remind here that, firstly, “\” (backslash) must be used in order to let the including of the definition given in this file into the other intrinsic commands correctly. Secondly, the flexible-kept parameter: *AX* will be read directly from the users’ initial (simulation/data analysis) setup set earlier on the website.

B.2.4 Helm form factor $F_{\text{Helm}}(Q)$

For the Helm form factor $F_{\text{Helm}}(Q)$, the nuclear skin thickness s , the radii r_0 and $R_A(A)$, and the effective nuclear radius $R_1(A)$ are defined as

Code A52: Nuclear skin thickness s

```
double ss_Helm = 0.9 * fm_U;
```

Code A53: r_0

```
double r_0_Helm = 0.52 * fm_U;
```

Code A54: $R_A(A)$

```
double R_A_Helm(int A)
{
  return (1.23 * cbrt(A) - 0.6) * fm_U;
}
```

Then the effective nuclear radius R_1 has to be redefined by

Code A55: Effective nuclear radius $R_1(A)$

```
double R_1_Helm(int A)
{
    return
        sqrt( R_A_Helm(A) * R_A_Helm(A)
            + (7.0 / 3.0) * (M_PI * M_PI) * (r_0_Helm * r_0_Helm)
            - 5.0 * ss_Helm * ss_Helm);
}
```

Therefore, similar to Codes A41 and A42 for the (derivative of the) Woods–Saxon form factor $F_{\text{WS}}^2(Q)$, we can define

Code A56: Squared Helm form factor $F_{\text{Helm}}^2(A, Q)$

```
double FQ_Helm(int A, double QQ)
{
    if (QQ == 0.0)
    {
        return 1.0;
    }

    else
    {
        return
            ( 3.0 * sphBesselj(1, qq(A, QQ) * R_1_Helm(A)) / (qq(A, QQ) * R_1_Helm(A)) ) *
            ( 3.0 * sphBesselj(1, qq(A, QQ) * R_1_Helm(A)) / (qq(A, QQ) * R_1_Helm(A)) ) *
            exp(-(qq(A, QQ) * ss_Helm) * (qq(A, QQ) * ss_Helm));
    }
}
```

and

Code A57: Derivative of the squared Helm form factor $dF_{\text{Helm}}^2(A, Q)/dQ$

```
double dFQdQ_Helm(int A, double QQ)
{
    if (QQ == 0.0)
    {
        return
            -0.4 *
            ( R_A_Helm(A) * R_A_Helm(A)
            + (7.0 / 3.0) * (M_PI * M_PI) * (r_0_Helm * r_0_Helm) ) *
            m_N(A);
    }

    else
    {
        return
            ( dsphBesselj(1, qq(A, QQ) * R_1_Helm(A)) * R_1_Helm(A) /
            sphBesselj(1, qq(A, QQ) * R_1_Helm(A))
            - 1.0 / qq(A, QQ)
            - (qq(A, QQ) * ss_Helm * ss_Helm) ) *
            FQ_Helm(A, QQ) * ((2.0 * m_N(A)) / qq(A, QQ));
    }
}
```

Note here that the expression (17) for R_1 has to be used.

On the other hand, for drawing the output generating WIMP–signal recoil spectrum, we need

Code A58: Squared Helm form factor $F_{\text{Helm}}^2(Q)$ (for Gnuplot)

```

M_PI = 3.141593

c    = 1.0
m_U  = 1e6 / (c * c)
fm_U = c / (0.197327 * 1e6)

m_N  = (0.938272 * m_U) * AX * 0.99

ss_Helm = 0.9 * fm_U
r_0_Helm = 0.52 * fm_U

R_A_Helm = ( 1.23 * (AX ** (1.0 / 3.0)) - 0.6 ) * fm_U

R_1_Helm
= sqrt( R_A_Helm * R_A_Helm
      + (7.0 / 3.0) * (M_PI ** 2.0) * (r_0_Helm * r_0_Helm)
      - 5.0 * ss_Helm * ss_Helm )

qq(x)
= sqrt(2.0 * m_N * x)

sphBesselj_1(x)
= ( sin(qq(x) * R_1_Helm) / (qq(x) * R_1_Helm) - cos(qq(x) * R_1_Helm) ) /
  (qq(x) * R_1_Helm)

FQ_Helm(x)
= ( 3.0 * sphBesselj_1(x) / (qq(x) * R_1_Helm) ) *
  ( 3.0 * sphBesselj_1(x) / (qq(x) * R_1_Helm) ) *
  exp(-(qq(x) * ss_Helm) * (qq(x) * ss_Helm))

```

Remind here that, firstly, “\” (backslash) must be used in order to let the including of the definition given in this file into the other intrinsic commands correctly. Secondly, the flexible–kept parameter: AX will be read directly from the users’ initial (simulation/data analysis) setup set earlier on the website.

B.2.5 Thin–shell form factor $F_{\text{TS}}(Q)$

For the thin–shell form factor $F_{\text{TS}}(Q)$, the lower and upper bounds of qR_1 given in Eq. (21), between which the form factor is a constant ($\simeq 0.047$), is defined by

Code A59: Lower and upper bounds of qR_1

```

qqR_1_min = 2.55;
qqR_1_max = 4.50;

```

and the constant is redefined as

Code A60: Constan of the thin-shell form factort ($\simeq 0.047$)

```
FQ_TS_const
= sphBesselj(0, qqR_1_min) *
  sphBesselj(0, qqR_1_min);
```

Meanwhile, the lower and upper energy bounds can be estimated by

Code A61: Lower and upper energy bounds

```
double QQ_SD_min(int A)
{
  return (qqR_1_min * qqR_1_min) / (2.0 * m_N(A) * R_1(A) * R_1(A));
}

double QQ_SD_max(int A)
{
  return (qqR_1_max * qqR_1_max) / (2.0 * m_N(A) * R_1(A) * R_1(A));
}
```

Then we can define

Code A62: Squared thin-shell form factor $F_{\text{TS}}^2(A, Q)$

```
double FQ_TS(int A, double QQ)
{
  if (QQ == 0.0)
  {
    return 1.0;
  }

  else
  if (QQ <= QQ_SD_min(A) ||
      QQ >= QQ_SD_max(A) )
  {
    return
      sphBesselj(0, qq(A, QQ) * R_1(A)) *
      sphBesselj(0, qq(A, QQ) * R_1(A));
  }

  else
  {
    return FQ_TS_const;
  }
}
```

For defining the derivative of the squared thin-shell form factor, $dF_{\text{TS}}^2(Q)/dQ$, first, we find that

$$\frac{dF_{\text{TS}}^2(Q)}{dQ} = \frac{d}{dQ} \left[j_0^2(qR_1) \right] = \left[\frac{j_0'(qR_1) R_1}{j_0(qR_1)} \right] \left(\frac{2m_N}{q} \right) F_{\text{TS}}^2(Q), \quad (\text{A27})$$

where we have used Eq. (A14). For the limit case $Q = q = 0$, since [42]

$$j_0(x) = \sqrt{\frac{\pi}{2x}} J_{1/2}(x) = \frac{\sin x}{x}, \quad (\text{A28})$$

we can get

$$j'_0(x) = \frac{x \cos x - \sin x}{x^2}. \quad (\text{A29})$$

Then, by letting $x = qR_1$, the term in the bracket of the right-hand side of Eq. (A27) (multiply by $1/q$) can be written and found that

$$\frac{j'_0(x)}{j_0(x)} \left(\frac{R_1^2}{x} \right) = \left(\frac{x \cos x - \sin x}{x^2 \sin x} \right) R_1^2 \rightarrow -\frac{R_1^2}{3}, \quad (\text{A30})$$

as $x \rightarrow 0$. Hence, we can obtain that, for the limit case $Q = q = 0$,

$$\left. \frac{dF_{\text{TS}}^2(Q)}{dQ} \right|_{Q=0} = -\left(\frac{2}{3} \right) R_1^2 m_N, \quad (\text{A31})$$

where we have used

$$F_{\text{TS}}^2(Q = 0) = 1. \quad (\text{A32})$$

Now, we can define the derivative of the squared thin-shell form factor as

Code A63: Derivative of the squared thin-shell form factor $dF_{\text{TS}}^2(A, Q)/dQ$

```
double dFQdQ_TS(int A, double QQ)
{
    if (QQ == 0.0)
    {
        return
            -(2.0 / 3.0) * (R_1(A) * R_1(A)) * m_N(A);
    }

    else
    if (QQ <= QQ_SD_min(A) ||
        QQ >= QQ_SD_max(A) )
    {
        return
            dsphBesselj(0, qq(A, QQ) * R_1(A)) * R_1(A) *
            sphBesselj(0, qq(A, QQ) * R_1(A)) *
            ( (2.0 * m_N(A)) / qq(A, QQ) );
    }

    else
    {
        return 0.0;
    }
}
```

Finally, for drawing the output generating WIMP-signal recoil spectrum, we need

Code A64: Squared thin-shell form factor $F_{TS}^2(Q)$ (for Gnuplot)

```

qqR_1_min = 2.55
qqR_1_max = 4.50

FQ_TS_const \
= (sin(qqR_1_min) / qqR_1_min) * \
  (sin(qqR_1_max) / qqR_1_max)

c = 1.0
m_U = 1e6 / (c * c)
fm_U = c / (0.197327 * 1e6)

m_N = (0.938272 * m_U) * AX * 0.99

ss = fm_U
R_A = 1.2 * (AX ** (1.0 / 3.0)) * fm_U
R_1 = sqrt(R_A * R_A - 5.0 * ss * ss)

qq(x) \
= sqrt(2.0 * m_N * x)

sphBesselj_0(x) \
= sin(qq(x) * R_1) / (qq(x) * R_1)

Xi(x) \
= ( (qq(x) * R_1 - qqR_1_min) / abs(qq(x) * R_1 - qqR_1_min) ) * \
  ( (qq(x) * R_1 - qqR_1_max) / abs(qq(x) * R_1 - qqR_1_max) ) )

FQ_TS(x) \
= sphBesselj_0(x) * sphBesselj_0(x) * \
  (1.0 + Xi(x)) / 2.0 \
+ FQ_TS_const * (1.0 - Xi(x)) / 2.0

```

Here we define an auxiliary function

$$\Xi(x) = \frac{x - x_{\min}}{|x - x_{\min}|} \cdot \frac{x - x_{\max}}{|x - x_{\max}|} = \begin{cases} +1, & \text{for } x \leq x_{\min} \text{ or } x \geq x_{\max}, \\ -1, & \text{for } x_{\min} \leq x \leq x_{\max}. \end{cases} \quad (\text{A33})$$

Thus we can get

$$\frac{1}{2}[1 + \Xi(x)] = \begin{cases} 1, & \text{for } x \leq x_{\min} \text{ or } x \geq x_{\max}, \\ 0, & \text{for } x_{\min} \leq x \leq x_{\max}, \end{cases} \quad (\text{A34a})$$

and

$$\frac{1}{2}[1 - \Xi(x)] = \begin{cases} 0, & \text{for } x \leq x_{\min} \text{ or } x \geq x_{\max}, \\ 1, & \text{for } x_{\min} \leq x \leq x_{\max}. \end{cases} \quad (\text{A34b})$$

Remind here that, firstly, “\” (backslash) must be used in order to let the including of the definition given in this file into the other intrinsic commands correctly. Secondly, the flexible-kept parameter: **AX** will be read directly from the users’ initial (simulation/data analysis) setup set earlier on the website.

B.3 Background spectrum

In this subsection, we give the codes for intrinsically defined simple artificial background spectrum given in Sec. 3.5 in the **AMIDAS** package.

B.3.1 Signal and background windows

Similar to the function $\Xi(x)$ defined in Eqs. (A33), (A34a) and (A34b), we define the signal and background windows and vacua for drawing the predicted WIMP-signal and artificial background spectra on the output plots together as following:

Code A65: Signal window

```
sgwindow(x) \
= ( 1 \
  - ( (x - Qmin_gen) / abs(x - Qmin_gen) ) * \
    ( (x - Qmax_gen) / abs(x - Qmax_gen) ) ) / 2.0
```

Code A66: Signal vacuum

```
sgvacuum(x) \
= ( 1 \
  + ( (x - Qmin_gen) / abs(x - Qmin_gen) ) * \
    ( (x - Qmax_gen) / abs(x - Qmax_gen) ) ) / 2.0
```

Code A67: Background window

```
bgwindow(x) \
= ( 1 \
  - ( (x - Qmin_bgwindow) / abs(x - Qmin_bgwindow) ) * \
    ( (x - Qmax_bgwindow) / abs(x - Qmax_bgwindow) ) ) / 2.0
```

Code A68: Background vacuum

```
bgvacuum(x) \
= ( 1 \
  + ( (x - Qmin_bgwindow) / abs(x - Qmin_bgwindow) ) * \
    ( (x - Qmax_bgwindow) / abs(x - Qmax_bgwindow) ) ) / 2.0
```

According to these definitions, **AMIDAS** will use the given analytic expressions to draw the WIMP-signal and background spectra exactly in the energy range between the minimal and maximal cut-offs $Q_{(\min, \max)}$ and $Q_{\text{bg}, (\min, \max)}$, respectively. Note here that the minimal and maximal cut-offs: `Qmin_gen`, `Qmax_gen` and `Qmin_bgwindow`, `Qmax_bgwindow` will be read directly from the users' initial (simulation/data analysis) setup set earlier on the website. Remind also here that “\” (backslash) must be used in order to let the including of the definition(s) given in this file into the other intrinsic commands correctly.

B.3.2 Constant background spectrum $(dR/dQ)_{\text{bg}, \text{const}}$

The simplest considerable (artificial) background spectrum is the constant one $(dR/dQ)_{\text{bg}, \text{const}}$ given in Eq. (23) [23]:

Code A69: Constant background spectrum $(dR/dQ)_{\text{bg,const}}(Q)$

```
double dRdQ_bg_const(double QQ)
{
    return 1.0;
}
```

For drawing the output generating background spectrum, we have

Code A70: Constant background spectrum $(dR/dQ)_{\text{bg,const}}(Q)$ (for Gnuplot)

```
dRdQ_bg_const(x) \
= 1.0
```

combined with the integral over $(dR/dQ)_{\text{bg,const}}(Q)$ for *normalizing* the background spectrum properly according to the user’s required background ratio (to the WIMP–induced signals):

Code A71: Integral over the constant background spectrum $\int (dR/dQ)_{\text{bg,const}}(Q) dQ$ (for Gnuplot)

```
IntdRdQ_bg_const(x) \
= x
```

Remind als here that “\” (backslash) must be used in order to let the including of the definition(s) given in this file into the other intrinsic commands correctly.

B.3.3 Exponential background spectrum $(dR/dQ)_{\text{bg,ex}}$

More realistically, we considered in our simulations the “target–dependent” exponential background spectrum $(dR/dQ)_{\text{bg,ex}}$ given in Eq. (24) [23]:

Code A72: Target–dependent exponential background spectrum $(dR/dQ)_{\text{bg,ex}}(A, Q)$

```
double Q_0_bg(int A)
{
    return pow(A, 0.6);
}

double dRdQ_bg_ex(int A, double QQ)
{
    return exp(-QQ / Q_0_bg(A));
}
```

For drawing the output generating background spectrum, one needs

Code A73: Target–dependent exponential background spectrum $(dR/dQ)_{\text{bg,ex}}(Q)$ (for Gnuplot)

```
dRdQ_bg_ex(x) \
= exp(-x / AX ** 0.6)
```

and

Code A74: Integral over the Target–dependent exponential background spectrum $\int (dR/dQ)_{\text{bg,ex}}(Q) dQ$ (for Gnuplot)

```
IntdRdQ_bg_ex(x) \
=-- (AX ** 0.6) * exp(-x / AX ** 0.6)
```

since

$$\int \left(\frac{dR}{dQ} \right)_{\text{bg,ex}} dQ = \int e^{-Q/A^{0.6}} dQ = -A^{0.6} e^{-Q/A^{0.6}}. \quad (\text{A35})$$

Remind that firstly, “\” (backslash) must be used in order to let the including of the definition(s) given in this file into the other intrinsic commands correctly. Secondly, the flexible-kept parameter: **AX** will be read directly from the users’ initial (simulation/data analysis) setup set earlier on the website.

B.3.4 Gaussian-excess background spectrum $(dR/dQ)_{\text{bg,Gau}}$

For describing background events induced by e.g. radioactivity of some nuclei, we introduce in our simulations the Gaussian-excess background spectrum $(dR/dQ)_{\text{bg,Gau}}$ given in Eq. (25) [39]:

Code A75: Gaussian-excess background spectrum $(dR/dQ)_{\text{bg,Gau}}(Q_{\text{bg,peak}}, \sigma_{Q,\text{bg}}, Q)$

```
double dRdQ_bg_Gau(double dRdQ_bg_mu, double dRdQ_bg_sigma_mu, double QQ)
{
    return
        exp(-(QQ - dRdQ_bg_mu)      *
            (QQ - dRdQ_bg_mu)      /
            (2.0                    *
            dRdQ_bg_sigma_mu *
            dRdQ_bg_sigma_mu ) ) /
        (dRdQ_bg_sigma_mu * sqrt(2.0 * M_PI));
}
```

For drawing the output generating background spectrum, one needs

Code A76: Gaussian-excess background spectrum $(dR/dQ)_{\text{bg,Gau}}(Q)$ (for Gnuplot)

```
M_PI = 3.141593

dRdQ_bg_mu      = 50.0
dRdQ_bg_sigma_mu = 2.0

dRdQ_bg_Gau(x)
= exp(-(x - dRdQ_bg_mu)      *
    (x - dRdQ_bg_mu)      /
    (2.0                    *
    dRdQ_bg_sigma_mu *
    dRdQ_bg_sigma_mu ) ) /
    (dRdQ_bg_sigma_mu * sqrt(2.0 * M_PI))
```

and

Code A77: Integral over the Gaussian-excess background spectrum $\int (dR/dQ)_{\text{bg,Gau}}(Q) dQ$ (for Gnuplot)

```
IntdRdQ_bg_Gau(x)
= (1.0 / 2.0) *
    erf( (x - dRdQ_bg_mu)      /
        (sqrt(2.0) * dRdQ_bg_sigma_mu ) )
```

since

$$\begin{aligned} \int \left(\frac{dR}{dQ} \right)_{\text{bg,Gau}} dQ &= \int \frac{1}{\sqrt{2\pi} \sigma_{Q,\text{bg}}} e^{-(Q-Q_{\text{bg,peak}})^2 / 2\sigma_{Q,\text{bg}}^2} dQ \\ &= \frac{1}{2} \operatorname{erf} \left(\frac{Q - Q_{\text{bg,peak}}}{\sqrt{2} \sigma_{Q,\text{bg}}} \right). \end{aligned} \quad (\text{A36})$$

Remind that firstly, “\” (backslash) must be used in order to let the including of the definition(s) given in this file into the other intrinsic commands correctly. Secondly, the flexible-kept parameters: `dRdQ_bg_mu` and `dRdQ_bg_sigma_mu` will be read directly from the users’ initial (simulation/data analysis) setup set earlier on the website.

C Intrinsically defined functions for Bayesian analyses

C.1 Fitting one-dimensional velocity distribution function

In this subsection, we give first the codes for intrinsically defined fitting velocity distribution functions given in Sec. 5.1 in the AMIDAS-II package.

C.1.1 Simple Maxwellian velocity distribution $f_{1,\text{Gau}}(v)$

From the expression (1) for the simple Maxwellian velocity distribution function $f_{1,\text{Gau}}(v)$, we can obtain that

$$\int f_{1,\text{Gau}}(v) dv = \operatorname{erf} \left(\frac{v}{v_0} \right) - \frac{2}{\sqrt{\pi}} \left(\frac{v}{v_0} \right) e^{-v^2/v_0^2}. \quad (\text{A37})$$

Then we can define

Code A78: Integral over the simple Maxwellian velocity distribution $\int f_{1,\text{Gau}}(v) dv$

```
double Int_f1v_Bayesian_fit_Gau(double vv, double aa, double bb, double cc)
{
    return
        erf(vv / aa)
        - (2.0 / sqrt(M_PI)) *
        (vv / aa) *
        exp(-(vv * vv) / (aa * aa));
}
```

and the fitting simple Maxwellian velocity distribution, which satisfies the normalization condition:

$$\int_0^{v_{\text{max}}} f_{1,\text{fit}}(v) dv = 1, \quad (\text{A38})$$

Code A79: Fitting simple Maxwellian velocity distribution $f_{1,\text{Gau}}(v)$

```
double f1v_Bayesian_fit_Gau(double vv, double aa, double bb, double cc, double N_f)
{
    return
        ( (4.0 / sqrt(M_PI)) *
          ((vv * vv) / (aa * aa * aa)) / v_U *
          exp(-(vv * vv) / (aa * aa)) ) /
        ( Int_f1v_Bayesian_fit_Gau(v_max, aa, bb, cc)
          - Int_f1v_Bayesian_fit_Gau(0.0, aa, bb, cc) );
}
```

On the other hand, for drawing output plots, we have

Code A80: Fitting simple Maxwellian velocity distribution $f_{1,\text{Gau}}(v)$ (for Gnuplot)

```
M_PI = 3.141593

f1v_Bayesian_fit_Gau(x)          \
= (4.0 / sqrt(M_PI))             * \
  ((x * x) / (a_Bayesian_fit * a_Bayesian_fit * a_Bayesian_fit)) * \
  exp(-(x * x) / (a_Bayesian_fit * a_Bayesian_fit)) / \
N_f
```

Here the normalization constant N_f will be estimated by using the function `Int_f1v_Bayesian_fit_Gau` defined in Code A78 as

$$N_f = \text{Int_f1v_Bayesian_fit_Gau}(v_{\max}, a_{\text{Bayesian_fit}}, 0, 0) - \text{Int_f1v_Bayesian_fit_Gau}(0, a_{\text{Bayesian_fit}}, 0, 0), \quad (\text{A39})$$

and the value of `a_Bayesian_fit` will be given from the fitting result. Remind that “\” (backslash) must be used in order to let the including of the definition(s) given in this file into the other intrinsic commands correctly.

C.1.2 Modified Maxwellian velocity distribution $f_{1,\text{Gau},k}(v)$

Since the power index k in the modified Maxwellian velocity distribution function $f_{1,\text{Gau},k}(v)$ is one of our fitting parameter and thus unfixed, an analytic form for the integral over $f_{1,\text{Gau},k}(v)$ is in general “unknown”. For this need, in the AMIDAS-II code one can simply define that

Code A81: Integral over the modified Maxwellian velocity distribution $\int f_{1,\text{Gau},k}(v) dv$

```
double Int_f1v_Bayesian_fit_Gau_k(double vv, double aa, double bb, double cc)
{
  return -1.0;
}
```

and

Code A82: Fitting modified Maxwellian velocity distribution $f_{1,\text{Gau},k}(v)$

```
double f1v_Bayesian_fit_Gau_k(double vv, double aa, double bb, double cc, double N_f)
{
  return
    ( (vv * vv) *
      pow( exp(-(vv * vv) / (cc * aa * aa)) - exp(-(v_max * v_max) / (cc * aa * aa)),
          cc ) ) /
  N_f;
}
```

Here the normalization constant N_f will be estimated *every time* by numerical integral over the function `f1v_Bayesian_fit_Gau_k` itself as as

$$N_f = \left[\int_0^{v_{\max}} f1v_Bayesian_fit_Gau_k(v, aa, bb, cc, 1) dv \right]^{-1}, \quad (\text{A40})$$

for different (scanning) points (aa, bb, cc) .

Similarly, for drawing the output plots, we have

Code A83: Fitting modified Maxwellian velocity distribution $f_{1,\text{Gau},k}(v)$ (for Gnuplot)

```

f1v_Bayesian_fit_Gau_k(x)
= (x * x)
  ( exp(-(x      * x      )
      (c_Bayesian_fit *
        a_Bayesian_fit * a_Bayesian_fit) )
    - exp(-(v_max * v_max)
      (c_Bayesian_fit *
        a_Bayesian_fit * a_Bayesian_fit) ) ) **
c_Bayesian_fit
N_f

```

Here the normalization constant N_f will be estimated by numerical integral over the function $f1v_Bayesian_fit_Gau_k$ defined in Code A82 as

$$N_f = \left[\int_0^{v_{\max}} f1v_Bayesian_fit_Gau_k(x, a_Bayesian_fit, 0, c_Bayesian_fit, 1) dx \right]^{-1}, \quad (\text{A41})$$

and the values of $a_Bayesian_fit$ and $c_Bayesian_fit$ will be given from the fitting result. Remind that “\” (backslash) must be used in order to let the including of the definition(s) given in this file into the other intrinsic commands correctly.

C.1.3 One-parameter shifted Maxwellian velocity distribution $f_{1,\text{sh},v_0}(v)$

From the expression (3) for the shifted Maxwellian velocity distribution function $f_{1,\text{sh}}(v)$, we can obtain that

$$\int f_{1,\text{sh}}(v) dv = \frac{1}{2} \left[\text{erf} \left(\frac{v + v_e}{v_0} \right) + \text{erf} \left(\frac{v - v_e}{v_0} \right) \right] + \frac{1}{2\sqrt{\pi}} \left(\frac{v_0}{v_e} \right) \left[e^{-(v+v_e)^2/v_0^2} - e^{-(v-v_e)^2/v_0^2} \right]. \quad (\text{A42})$$

Then, by using Eq. (27), we can define

Code A84: Integral over the one-parameter shifted Maxwellian velocity distribution $\int f_{1,\text{sh},v_0}(v) dv$

```

double Int_f1v_Bayesian_fit_sh_v0(double vv, double aa, double bb, double cc)
{
  return
    (1.0 / 2.0) *
    ( erf((vv + (aa * 1.05)) / aa)
      + erf((vv - (aa * 1.05)) / aa) )
    + (1.0 / 2.0 / sqrt(M_PI)) *
    (aa / (aa * 1.05)) *
    ( exp(-(vv + (aa * 1.05)) * (vv + (aa * 1.05)) / (aa * aa))
      - exp(-(vv - (aa * 1.05)) * (vv - (aa * 1.05)) / (aa * aa)) );
}

```

and the fitting one-parameter shifted Maxwellian velocity distribution, which satisfies the normalization condition (A38),

Code A85: Fitting one-parameter shifted Maxwellian velocity distribution $f_{1,\text{sh},v_0}(v)$

```
double flv_Bayesian_fit_sh_v0(double vv, double aa, double bb, double cc, double N_f)
{
    return
        ( (1.0 / sqrt(M_PI)) *
          (vv / aa / (aa * 1.05)) / v_U *
          ( exp(-(vv - (aa * 1.05)) * (vv - (aa * 1.05)) / (aa * aa) )
            - exp(-(vv + (aa * 1.05)) * (vv + (aa * 1.05)) / (aa * aa) ) ) ) /
        ( Int_flv_Bayesian_fit_sh_v0(v_max, aa, bb, cc)
          - Int_flv_Bayesian_fit_sh_v0(0.0, aa, bb, cc) );
}
```

On the other hand, for drawing output plots, we have

Code A86: Fitting one-parameter shifted Maxwellian velocity distribution $f_{1,\text{sh},v_0}(v)$ (for Gnuplot)

```
M_PI = 3.141593

flv_Bayesian_fit_sh_v0(x) \
= (1.0 / sqrt(M_PI)) * \
(x / a_Bayesian_fit / (a_Bayesian_fit * 1.05)) * \
( exp(-(x - (a_Bayesian_fit * 1.05)) * \
      (x - (a_Bayesian_fit * 1.05)) / \
      (a_Bayesian_fit * a_Bayesian_fit) ) \
- exp(-(x + (a_Bayesian_fit * 1.05)) * \
      (x + (a_Bayesian_fit * 1.05)) / \
      (a_Bayesian_fit * a_Bayesian_fit) ) ) / \
N_f
```

Here the normalization constant N_f will be estimated by using the function `Int_flv_Bayesian_fit_sh_v0` defined in Code A84 as

$$N_f = \text{Int_flv_Bayesian_fit_sh_v0}(v_{\text{max}}, a_{\text{Bayesian_fit}}, 0, 0) - \text{Int_flv_Bayesian_fit_sh_v0}(0, a_{\text{Bayesian_fit}}, 0, 0), \quad (\text{A43})$$

and the value of `a_Bayesian_fit` will be given from the fitting result. Remind that “\” (backslash) must be used in order to let the including of the definition(s) given in this file into the other intrinsic commands correctly.

C.1.4 Shifted Maxwellian velocity distribution $f_{1,\text{sh}}(v)$

According to Eq. (A42), we can define

Code A87: Integral over the shifted Maxwellian velocity distribution $\int f_{1,\text{sh}}(v) dv$

```
double Int_f1v_Bayesian_fit_sh(double vv, double aa, double bb, double cc)
{
    return
        (1.0 / 2.0) *
        ( erf((vv + bb) / aa)
          + erf((vv - bb) / aa) )
    + (1.0 / 2.0 / sqrt(M_PI)) *
        (aa / bb) *
        ( exp(-(vv + bb) * (vv + bb) / (aa * aa))
          - exp(-(vv - bb) * (vv - bb) / (aa * aa)) );
}
```

and the fitting shifted Maxwellian velocity distribution, which satisfies the normalization condition (A38),

Code A88: Fitting shifted Maxwellian velocity distribution $f_{1,\text{sh}}(v)$

```
double f1v_Bayesian_fit_sh(double vv, double aa, double bb, double cc, double N_f)
{
    return
        ( (1.0 / sqrt(M_PI))
          (vv / aa / bb) / v_U
          ( exp(-(vv - bb) * (vv - bb) / (aa * aa))
            - exp(-(vv + bb) * (vv + bb) / (aa * aa)) ) ) ) /
        ( Int_f1v_Bayesian_fit_sh(v_max, aa, bb, cc)
          - Int_f1v_Bayesian_fit_sh(0.0, aa, bb, cc) );
}
```

On the other hand, for drawing output plots, we have

Code A89: Fitting shifted Maxwellian velocity distribution $f_{1,\text{sh}}(v)$ (for Gnuplot)

```
M_PI = 3.141593

f1v_Bayesian_fit_sh(x) \
= (1.0 / sqrt(M_PI)) * \
(x / a_Bayesian_fit / b_Bayesian_fit) * \
( exp(-(x - b_Bayesian_fit) * \
    (x - b_Bayesian_fit) / \
    (a_Bayesian_fit * a_Bayesian_fit)) \
- exp(-(x + b_Bayesian_fit) * \
    (x + b_Bayesian_fit) / \
    (a_Bayesian_fit * a_Bayesian_fit)) ) ) / \
N_f
```

Here the normalization constant N_f will be estimated by using the function `Int_f1v_Bayesian_fit_sh` defined in Code A87 as

$$N_f = \text{Int_f1v_Bayesian_fit_sh}(v_{\text{max}}, a_{\text{Bayesian_fit}}, b_{\text{Bayesian_fit}}, 0) \\ - \text{Int_f1v_Bayesian_fit_sh}(0, a_{\text{Bayesian_fit}}, b_{\text{Bayesian_fit}}, 0), \quad (\text{A44})$$

and the values of `a_Bayesian_fit` and `b_Bayesian_fit` will be given from the fitting result. Remind that “\” (backslash) must be used in order to let the including of the definition(s) given in this file into the other intrinsic commands correctly.

C.1.5 Variated shifted Maxwellian velocity distribution $f_{1,\text{sh},\Delta v}(v)$

According to Eq. (A42) and the relation given in Eq. (29), we can define

Code A90: Integral over the variated shifted Maxwellian velocity distribution $\int f_{1,\text{sh},\Delta v}(v) dv$

```
double Int_f1v_Bayesian_fit_sh_Dv(double vv, double aa, double bb, double cc)
{
    return
        (1.0 / 2.0) *
        ( erf((vv + aa + bb) / aa)
          + erf((vv - aa - bb) / aa) )
    + (1.0 / 2.0 / sqrt(M_PI)) *
        (aa / (aa + bb)) *
        ( exp(-(vv + bb) * (vv + aa + bb) / (aa * aa) )
          - exp(-(vv - bb) * (vv - aa - bb) / (aa * aa) ) );
}
```

and the fitting variated shifted Maxwellian velocity distribution, which satisfies the normalization condition (A38),

Code A91: Fitting variated shifted Maxwellian velocity distribution $f_{1,\text{sh},\Delta v}(v)$

```
double f1v_Bayesian_fit_sh_Dv(double vv, double aa, double bb, double cc, double N_f)
{
    return
        ( (1.0 / sqrt(M_PI))
          (vv / aa / (aa + bb)) / v_U
          ( exp(-(vv - aa - bb) * (vv - aa - bb) / (aa * aa) )
            - exp(-(vv + aa + bb) * (vv + aa + bb) / (aa * aa) ) ) ) /
        ( Int_f1v_Bayesian_fit_sh_Dv(v_max, aa, bb, cc)
          - Int_f1v_Bayesian_fit_sh_Dv(0.0, aa, bb, cc) );
}
```

On the other hand, for drawing output plots, we have

Code A92: Fitting variated shifted Maxwellian velocity distribution $f_{1,\text{sh},\Delta v}(v)$ (for Gnuplot)

```
M_PI = 3.141593

f1v_Bayesian_fit_sh_Dv(x) \
= (1.0 / sqrt(M_PI)) * \
(x / a_Bayesian_fit / (a_Bayesian_fit + b_Bayesian_fit)) * \
( exp(-(x - a_Bayesian_fit - b_Bayesian_fit) * \
    (x - a_Bayesian_fit - b_Bayesian_fit) / \
    (a_Bayesian_fit * a_Bayesian_fit) ) \
- exp(-(x + a_Bayesian_fit + b_Bayesian_fit) * \
    (x + a_Bayesian_fit + b_Bayesian_fit) / \
    (a_Bayesian_fit * a_Bayesian_fit) ) ) / \
N_f
```

Here the normalization constant N_f will be estimated by using the function

`Int_f1v_Bayesian_fit_sh_Dv` defined in Code A90 as

$$N_f = \text{Int_f1v_Bayesian_fit_sh_Dv}(v_{\text{max}}, a_{\text{Bayesian_fit}}, b_{\text{Bayesian_fit}}, 0) \\ - \text{Int_f1v_Bayesian_fit_sh_Dv}(0, a_{\text{Bayesian_fit}}, b_{\text{Bayesian_fit}}, 0), \quad (\text{A45})$$

and the values of `a_Bayesian_fit` and `b_Bayesian_fit` will be given from the fitting result. Remind that “\” (backslash) must be used in order to let the including of the definition(s) given in this file into the other intrinsic commands correctly.

C.2 Distribution function for describing the statistical uncertainty

In this subsection, we give the codes for intrinsically defined distribution functions for describing the statistical uncertainty given in Sec. 5.2 in the **AMIDAS-II** package, which will in turn be used in the likelihood function in Bayesian analyses.

C.2.1 Poisson statistical–uncertainty distribution $\text{Poi}(x_i, y_i; y_{i,\text{th}})$

For describing, e.g. the (expected) signal/background event rates (numbers), one needs usually the Poisson statistical–uncertainty distribution $\text{Poi}(x_i, y_i; a_j, j = 1, 2, \dots, N_{\text{Bayesian}})$ given in Eq. (30). For this use, we define

Code A93: Poisson statistical–uncertainty distribution $\text{Poi}(x_i, y_i; y_{i,\text{th}})$

```
double Bayesian_DF_Poi(double y_th, double y_rec)
{
    if (y_rec == 0.0)
    {
        return
            exp(-y_th);
    }

    else
    if (y_rec > 0.0 &&
        y_rec <= 1.0 )
    {
        return
            pow(y_th, y_rec) / exp(lgamma(y_rec + 1.0)) *
            exp(-y_th);
    }

    else
    if (y_expt > 1.0)
    {
        return
            (y_th / y_rec) *
            Bayesian_DF_Poi(y_th, y_rec - 1.0);
    }
}
```

Here we have used the expression for the Poisson distribution with μ as the expectation value:

$$\text{Poi}(x; \mu) = \frac{\mu^x e^{-\mu}}{x!}, \quad \text{for } x \geq 0,$$

$$= \begin{cases} e^{-\mu}, & \text{for } x = 0, \\ \frac{\mu^x e^{-\mu}}{\Gamma(x+1)}, & \text{for } 0 < x \leq 1, \\ \left(\frac{\mu}{x}\right) \text{Poi}(x-1; \mu), & \text{for } 1 < x, \end{cases} \quad (\text{A46})$$

where the gamma function $\Gamma(x)$ is defined by

$$\Gamma(x) \equiv \int_0^\infty t^{x-1} e^{-t} dt = \frac{\Gamma(x+1)}{x}, \quad (\text{A47})$$

and `lgamma(x)` is the intrinsic C mathematical function defined as

$$\text{lgamma}(x) \equiv \ln \Gamma(x). \quad (\text{A48})$$

C.2.2 (Double-)Gaussian statistical-uncertainty distribution $\text{Gau}(x_i, y_i, y_{i,(\text{lo}, \text{hi})}; y_{i, \text{th}})$

For the most commonly needed Gaussian statistical-uncertainty distribution, we define in the AMIDAS-II package an *asymmetric* form:

Code A94: (Double-)Gaussian statistical-uncertainty distribution $\text{Gau}(x_i, y_i, y_{i,(\text{lo}, \text{hi})}; y_{i, \text{th}})$

```
double Bayesian_DF_Gau(double y_th, double y_rec, double y_lo, double y_hi)
{
    if (y_rec >= y_th)
    {
        return
            exp(-(y_rec - y_th) * (y_rec - y_th) /
                (2.0 * (y_rec - y_lo) * (y_rec - y_lo))) /
            (sqrt(2.0 * M_PI) * (y_rec - y_lo));
    }

    else
    if (y_rec < y_th)
    {
        return
            exp(-(y_rec - y_th) * (y_rec - y_th) /
                (2.0 * (y_hi - y_rec) * (y_hi - y_rec))) /
            (sqrt(2.0 * M_PI) * (y_hi - y_rec));
    }
}
```

Remind that, for the case that the analyzed/fitted data point is larger (smaller) than the theoretical value estimated from the fitting (WIMP velocity distribution) function, we take the 1σ lower (upper) (statistical) uncertainty as the uncertainty $\sigma(y_i)$ in Eq. (31).

C.3 Distribution function of each fitting parameter

In the AMIDAS-II package, we offer so far three probability distribution functions for each fitting parameter for describing the prior knowledge about it.

C.3.1 Flat probability distribution function $p_{i,\text{flat}}(a_i)$

For the case that a prior knowledge about one (of the) fitting parameter(s) is “unknown”, one can use the flat probability distribution function $p_{i,\text{flat}}(a_i)$ given in Eq. (32):

Code A95: Flat probability distribution function $p_{i,\text{flat}}(a_i)$

```
double Bayesian_DF_a_flat(double aa)
{
    return 1.0;
}
```

C.3.2 Poisson probability distribution function $p_{i,\text{Poi}}(a_i; \mu_{a,i})$

For the case that one (of the) fitting parameter(s) is expected to be with an uncertainty obeying the “Poisson” statistics, one can use the Poisson probability distribution function $p_{i,\text{Poi}}(a_i; \mu_{a,i})$ given in Eq. (33):

Code A96: Poisson probability distribution function $p_{i,\text{Poi}}(a_i; \mu_{a,i})$

```
double Bayesian_DF_a_Poi(double aa)
{
    if (aa == 0.0)
    {
        return
            exp(-a_ave_Bayesian);
    }

    else
    if (aa > 0.0 &&
        aa <= 1.0 )
    {
        return
            pow(a_ave_Bayesian, aa) / exp(lgamma(aa + 1.0)) *
            exp(-a_ave_Bayesian);
    }

    else
    if (aa > 1.0)
    {
        return
            (a_ave_Bayesian / aa) *
            Bayesian_DF_a_Poi(a_ave_Bayesian, aa - 1.0);
    }
}
```

C.3.3 Gaussian probability distribution function $p_{i,\text{Gau}}(a_i; \mu_{a,i}, \sigma_{a,i})$

For the most common case that one (of the) fitting parameter(s) is expected to be with an uncertainty obeying the “Gaussian/normal” distribution, one can use the Gaussian probability distribution function $p_{i,\text{Gau}}(a_i; \mu_{a,i}, \sigma_{a,i})$ given in Eq. (34):

Code A97: Gaussian probability distribution function $p_{i,\text{Gau}}(a_i; \mu_{a,i}, \sigma_{a,i})$

```
double Bayesian_DF_a_Gau(double aa)
{
    return
        exp(-(aa - a_ave_Bayesian) *
            (aa - a_ave_Bayesian) /
            (2.0 *
                sigma_a_ave_Bayesian *
                sigma_a_ave_Bayesian ) ) /
        (sigma_a_ave_Bayesian * sqrt(2.0 * M_PI));
}
```

References

- [1] G. Jungman, M. Kamionkowski and K. Griest, “*Supersymmetric Dark Matter*”, *Phys. Rep.* **267**, 195–373 (1996), [arXiv:hep-ph/9506380](#).
- [2] G. Bertone, D. Hooper and J. Silk, “*Particle Dark Matter: Evidence, Candidates and Constraints*”, *Phys. Rep.* **405**, 279–390 (2005), [arXiv:hep-ph/0404175](#).
- [3] L. Bergström, “*Dark Matter Evidence, Particle Physics Candidates and Detection Methods*”, *Ann. Phys.* **524**, 479–496 (2012), [arXiv:1205.4882 \[astro-ph.HE\]](#).
- [4] M. Drees and C.-L. Shan, “*Reconstructing the Velocity Distribution of Weakly Interacting Massive Particles from Direct Dark Matter Detection Data*”, *J. Cosmol. Astropart. Phys.* **0706**, 011 (2007), [arXiv:astro-ph/0703651](#).
- [5] C.-L. Shan, “*Bayesian Reconstruction of the Velocity Distribution of Weakly Interacting Massive Particles from Direct Dark Matter Detection Data*”, [arXiv:1403.xxxx \[astro-ph.HE\]](#) (2014).
- [6] M. Drees and C.-L. Shan, “*Model-Independent Determination of the WIMP Mass from Direct Dark Matter Detection Data*”, *J. Cosmol. Astropart. Phys.* **0806**, 012 (2008), [arXiv:0803.4477 \[hep-ph\]](#).
- [7] C.-L. Shan, “*Estimating the Spin-Independent WIMP–Nucleon Coupling from Direct Dark Matter Detection Data*”, [arXiv:1103.0481 \[hep-ph\]](#) (2011).
- [8] C.-L. Shan, “*Determining Ratios of WIMP–Nucleon Cross Sections from Direct Dark Matter Detection Data*”, *J. Cosmol. Astropart. Phys.* **1107**, 005 (2011), [arXiv:1103.0482 \[hep-ph\]](#).
- [9] <http://www-ilias.cea.fr/>.
- [10] <http://pisrv0.pit.physik.uni-tuebingen.de/darkmatter/index1.html>.
- [11] <http://pisrv0.pit.physik.uni-tuebingen.de/darkmatter/amidas/>.
- [12] <http://www.tir.tw/phys/hep/dm/amidas/>.
- [13] C.-L. Shan, “*The AMIDAS Website: An Online Tool for Direct Dark Matter Detection Experiments*”, *AIP Conf. Proc.* **1200**, 1031–1034 (2010), [arXiv:0909.1459 \[astro-ph.IM\]](#).

- [14] C.-L. Shan, “Uploading User-Defined Functions onto the **AMIDAS** Website”, arXiv:0910.1971 [astro-ph.IM] (2009).
- [15] <http://www.gnuplot.info/>.
- [16] D. R. Tovey *et al.*, “A New Model-Independent Method for Extracting Spin-Dependent Cross Section Limits from Dark Matter Searches”, *Phys. Lett. B* **488**, 17–26 (2000), arXiv:hep-ph/0005041.
- [17] F. Giuliani and T. A. Girard, “Model-Independent Limits from Spin-Dependent WIMP Dark Matter Experiments”, *Phys. Rev. D* **71**, 123503 (2005), arXiv:hep-ph/0502232.
- [18] T. A. Girard and F. Giuliani, “On the Direct Search for Spin-Dependent WIMP Interactions”, *Phys. Rev. D* **75**, 043512 (2007), arXiv:hep-ex/0511044.
- [19] M. T. Ressell and D. J. Dean, “Spin-Dependent Neutralino-Nucleus Scattering for $A \sim 127$ Nuclei”, *Phys. Rev. C* **56**, 535–546 (1997), arXiv:hep-ph/9702290.
- [20] P. Toivanen, M. Kortelainen, J. Suhonen and J. Toivanen, “Large-Scale Shell-Model Calculations of Elastic and Inelastic Scattering Rates of Lightest Supersymmetric Particles (LSP) on $I-127$, $Xe-129$, $Xe-131$, and $Cs-133$ nuclei”, *Phys. Rev. C* **79**, 044302 (2009).
- [21] M. Garny, A. Ibarra, M. Pato and S. Vogl, “On the Spin-Dependent Sensitivity of *XENON100*”, *Phys. Rev. D* **87**, 056002 (2013), arXiv:1211.4573 [hep-ph].
- [22] P. Klos, J. Menéndez, D. Gazit and A. Schwenk, “Large-Scale Nuclear Structure Calculations for Spin-Dependent WIMP Scattering with Chiral Effective Field Theory Currents”, *Phys. Rev. D* **88**, 083516 (2013), arXiv:1304.7684 [nucl-th].
- [23] Y.-T. Chou and C.-L. Shan, “Effects of Residue Background Events in Direct Dark Matter Detection Experiments on the Determination of the WIMP Mass”, *J. Cosmol. Astropart. Phys.* **1008**, 014 (2010), arXiv:1003.5277 [hep-ph].
- [24] C.-L. Shan, “Effects of Residue Background Events in Direct Dark Matter Detection Experiments on the Reconstruction of the Velocity Distribution Function of Halo WIMPs”, *J. Cosmol. Astropart. Phys.* **1006**, 029 (2010), arXiv:1003.5283 [astro-ph.HE].
- [25] C.-L. Shan, “Effects of Residue Background Events in Direct Dark Matter Detection Experiments on the Estimation of the Spin-Independent WIMP-Nucleon Coupling”, arXiv:1103.4049 [hep-ph] (2011).
- [26] C.-L. Shan, “Effects of Residue Background Events in Direct Dark Matter Detection Experiments on the Determinations of Ratios of WIMP-Nucleon Cross Sections”, arXiv:1104.5305 [hep-ph] (2011).
- [27] M. Lisanti, L. E. Strigari, J. G. Wacker and R. H. Wechsler, “The Dark Matter at the End of the Galaxy”, *Phys. Rev. D* **83**, 023519 (2011), arXiv:1010.4300 [astro-ph.CO].
- [28] Y.-Y. Mao, L. E. Strigari, R. H. Wechsler, H.-Y. Wu and O. Hahn, “Halo-to-Halo Similarity and Scatter in the Velocity Distribution of Dark Matter” *Astrophys. J.* **764**, 35 (2013), arXiv:1210.2721 [astro-ph.CO].

- [29] Y.-Y. Mao, L. E. Strigari and R. H. Wechsler, “*Connecting Direct Dark Matter Detection Experiments to Cosmologically Motivated Halo Models*”, [arXiv:1304.6401](#) [[astro-ph.CO](#)] (2013).
- [30] M. Kuhlen, A. Pillepich, J. Guedes and P. Madau, “*The Distribution of Dark Matter in the Milky Way’s Disk*”, [arXiv:1308.1703](#) [[astro-ph.GA](#)] (2013).
- [31] J. D. Lewin and P. F. Smith, “*Review of Mathematics, Numerical Factors, and Corrections for Dark Matter Experiments Based on Elastic Nuclear Recoil*”, *Astropart. Phys.* **6**, 87–112 (1996).
- [32] C.-L. Shan, “*Determining the Mass of Dark Matter Particles with Direct Detection Experiments*”, *New J. Phys.* **11**, 105013 (2009), [arXiv:0903.4320](#) [[hep-ph](#)].
- [33] S. P. Ahlen *et al.*, “*Limits on Cold Dark Matter Candidates from an Ultralow Background Germanium Spectrometer*”, *Phys. Lett. B* **195**, 603–608 (1987).
- [34] K. Freese, J. Frieman and A. Gould, “*Signal Modulation in Cold-Dark-Matter Detection*”, *Phys. Rev. D* **37**, 3388–3405 (1988).
- [35] J. Engel, “*Nuclear Form-Factors for the Scattering of Weakly Interacting Massive Particles*”, *Phys. Lett. B* **264**, 114–119 (1991).
- [36] G. Eder, “*Nuclear Forces*”, MIT Press, Chapter 7 (1968).
- [37] R. H. Helm, “*Inelastic and Elastic Scattering of 187-MeV Electrons from Selected Even-Even Nuclei*”, *Phys. Rev.* **104**, 1466–1475 (1956).
- [38] H. V. Klapdor-Kleingrothaus, I. V. Krivosheina and C. Tomei, “*New Limits on Spin-Dependent Weakly Interacting Massive Particle (WIMP) Nucleon Coupling*”, *Phys. Lett. B* **609**, 226–231 (2005).
- [39] C.-L. Shan, “*Analyzing Direct Dark Matter Detection Data with Unrejected Background Events on the **AMIDAS** Website*”, *J. Phys.: Conf. Ser.* **384**, 012006 (2012), [arXiv:1111.6346](#) [[hep-ph](#)].
- [40] C.-L. Shan, “*Analyzing Direct Dark Matter Detection Data on the **AMIDAS** Website*”, *EAS Publ. Ser.* **53**, 77–88 (2012), [arXiv:1109.0125](#) [[hep-ph](#)].
- [41] J. Beringer *et al.*, “*The Review of Particle Physics 2012*”, *Phys. Rev. D* **86**, 010001 (2012), 1. *Physical Constants*.
- [42] M. R. Spiegel, “*Schaum’s Mathematical Handbook of Formulas and Tables*”, McGraw-Hill, international edition (1990). Note that there should be a “–” (miuns) sign on the right-hand side of Eq. 24.26, namely,

$$J_{-3/2}(x) = -\sqrt{\frac{2}{\pi x}} \left(\frac{\cos x}{x} + \sin x \right). \quad (\text{A49})$$